



## Resiliencia y Optimización

**Fecha entrega:** No es de entrega obligatoria.

Una de las estrategias más habituales para brindar disponibilidad a los servicios y redes es la redundancia. Poseer réplicas de un componente permite reducir los puntos únicos de falla de una red, sea éste un enlace local, un servidor o servicio, un router, un enlace hacia Internet, la provisión de energía, etc. Algunas arquitecturas redundantes brindan, como beneficio adicional, una mejora en performance mediante la distribución de carga de trabajo en distintos recursos (servidores, enlaces, almacenamiento, etc).

Categorías ISO: **FCAPS**.

### Bibliografía

- OPPENHEIMER, P. 2011. *Top-Down Network Design (3da ed)*. CISCO Press.
  - Capítulo 5. Sección “Redundant Network Design Topologies” (pp. 130-132)
- LIMONCELLI, T., et al. 2017. *The Practice of System and Network Administration (3ra ed)*. Addison-Wesley Professional.
  - Capítulo 18. “The Service Resiliency and Performance Patterns” (pp. 321-333)
- LIMONCELLI, T., et al. 2014. *The Practice of Cloud System Administration*. Addison-Wesley Professional.
  - Capítulo 4. “Application Architectures” (pp. 69-85)

### Referencias

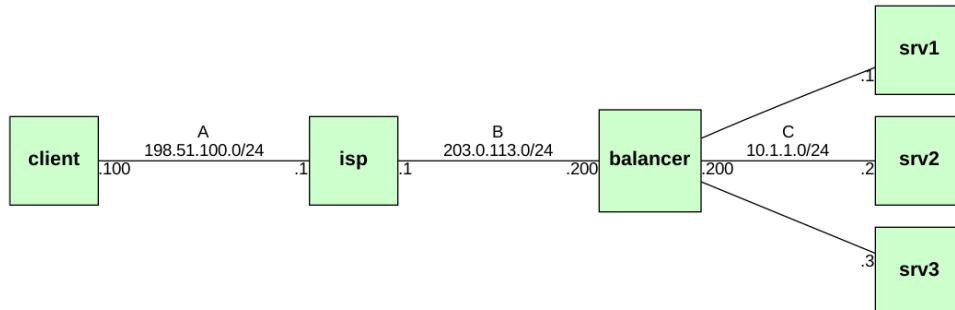
- HAProxy Starter Guide  
<https://cbonte.github.io/haproxy-dconv/2.0/intro.html>
- NGINX HTTP Load Balancing  
<https://docs.nginx.com/nginx/admin-guide/load-balancer/http-load-balancer/>
- NGINX and the “Power of Two Choices” Load-Balancing Algorithm  
<https://www.nginx.com/blog/nginx-power-of-two-choices-load-balancing-algorithm/>

### Trabajo Práctico

1. Realice una consulta con el comando `dig` al servidor `netflix.com` . Repita la operación varias veces. ¿La respuesta del servidor es siempre la misma? ¿Por qué?
2. Utilice el servicio DNS checker para consultar los dominios `www.unlu.edu.ar` y `www.google.com` . Este servicio informa los resultados que se obtienen desde distintos lugares del mundo. ¿Se observa alguna diferencia entre la respuesta para los diferentes nombres de dominio consultados?
3. El uso de múltiples Resource Records de tipo A para un mismo nombre de dominio, ¿a qué capa del modelo OSI brinda redundancia? ¿sirve para otros servicios?
4. Realice las experiencias en las próximas páginas y responda en el Trabajo Práctico las preguntas que figuran en los puntos de cada una de ellas.

## Balaneo de carga mediante Nginx

En esta experiencia se utilizará el lab [kathara-lab\\_nginx.tar.gz](http://kathara-lab_nginx.tar.gz) que posee la siguiente topología:



1. Descargue e inicie el laboratorio [kathara-lab\\_nginx.tar.gz](http://kathara-lab_nginx.tar.gz).
2. El equipo *balancer* es el servidor web de [www.empresa.com](http://www.empresa.com)  
En dicho equipo, inicie el servidor web apache ejecutando `service apache2 start`
3. En el cliente, ejecute el navegador web de consola `w3m` para obtener la web de la empresa:  
`w3m http://www.empresa.com/`
4. Salga del navegador pulsando la tecla `q` y luego la tecla `y`.
5. Utilizando `apachebench` (`ab`) obtenga métricas de performance de la arquitectura con servidor único (ver TP Herramientas de Diagnóstico de Redes).  
`ab -n 10000 -c 50 http://www.empresa.com/`
6. La empresa ficticia está por lanzar un nuevo producto y espera muchos accesos a su sitio web. Para evitar que se sature, ha contratado sus servicios para configurar un esquema de distribución de carga. Para ello, en el servidor *balancer*, detenga el servidor web existente:  
`service apache2 stop`
7. Asigne las direcciones a los servidores *srv1*, *srv2* y *srv3* según lo indicado en la topología. Este será el *pool* o *granja* de servidores.
8. En el equipo *balancer*, configure el servidor web *nginx* para que distribuya las peticiones HTTP que recibe entre los servidores del pool. Para ello:
  - a. Elimine la configuración existente en `/etc/nginx/sites-enabled/`  

```
cd /etc/nginx/sites-enabled
rm default
```
  - b. Utilizando `nano` o `mcedit`, cree un nuevo archivo de configuración llamado `balancer` y coloque el siguiente contenido:

```
server {
    listen 80;          # escuchar en puerto 80
    location / {       # reenviar todas las peticiones al pool
        proxy_pass http://pool;
    }
}
```



- c. En el mismo archivo de configuración (fuera de la definición de `server`) agregue el pool de servidores. En las sentencias `server` de dicho archivo, reemplace las letras por las direcciones IP de los servidores del pool para apuntar a los que usted ha definido y guarde el archivo.

```
upstream pool {
# direcciones ip de las replicas
server X.X.X.X:80 weight=3 max_fails=3 fail_timeout=10s;
server Y.Y.Y.Y:80;
server Z.Z.Z.Z:80;
# otros metodos de distribucion de carga (descomente para activar)
# ip_hash;
# least_conn;
}
```

- d. Inicie el servicio `nginx` y verifique que funciona haciendo `netstat -tnlp`

```
service nginx restart
netstat -tnlp
```

9. En el mismo servidor, inicie una captura mediante `tcpdump` en todas las interfaces:

```
tcpdump -i any -w /hosthome/captura-nginx.pcap
```

10. Ahora desde el equipo *cliente*, utilice `w3m` o `curl` para volver a consultar la página web:

```
curl http://www.empresa.com/
```

Las páginas son ligeramente diferentes (a propósito) pues al pie de cada una de ellas se añadió una línea para indicar qué servidor del pool respondió la petición.

Vuelva a realizar la consulta varias veces. ¿Responde siempre el mismo servidor del pool?

11. Detenga uno de los servidores del pool (*srv1*, *srv2* o *srv3*) escribiendo el comando `service apache2 stop` y vuelva a realizar la consulta. ¿Se perdió disponibilidad? Detenga otro servidor y repita la consulta. ¿Se perdió disponibilidad ahora? Repita el proceso. ¿Qué código de error retorna el servidor *balancer* ahora? ¿Ha visto un error similar en Internet?
12. Detenga la captura y guárdela para un análisis posterior.
13. Vuelva a iniciar los servidores web *srv1*, *srv2*, *srv3* con `service apache2 start`. Utilizando `apachebench` (`ab`) vuelva a obtener las métricas de performance y compárela con las de primera parte de la experiencia.
14. En cada uno de los servidores del pool ejecute `wc -l /var/log/apache2/access.log`. El archivo `access.log` contiene, en cada línea, un registro de cada uno de los accesos realizados al sitio web. Gracias al comando `wc -l` (que permite contar las líneas de un archivo) usted obtendrá el número de peticiones recibidas en cada servidor. Compare la salida de ese comando en cada uno de los servidores. ¿Se corresponden con lo esperado? ¿Por qué?
15. ¿Qué otros métodos de distribución de carga soporta Nginx? Describa brevemente cada uno de ellos.
16. ¿A qué capa del modelo OSI brinda redundancia esta arquitectura de Nginx? ¿Sirve para otros servicios?
17. Indique ¿cuál es el único punto de fallo (SPOF) de la arquitectura? ¿hay más de un SPOF?



## Alta disponibilidad mediante CARP / VRRP

Con la implementación de un balanceador de carga hemos logrado aumentar la disponibilidad del servicio, sin embargo aún podemos seguir mejorando esta arquitectura. Como has podido notar, el balanceador de carga se configura ahora como el punto de fallá único de nuestro diseño. Si este deja de funcionar, nuestro pool de servidores (por más grande que sea) quedará totalmente inútil. Aumentar la redundancia del balanceador nos ayudará, por lo tanto, a lidiar con este problema.

Para aumentar la disponibilidad de este diseño entonces agregaremos una copia del equipo *balancer* que operara en modo hot-standby, es decir, que agregaremos un segundo servidor que estará preparado para usarse al momento de que el balanceador de carga maestro deje de funcionar. Para implementar esta solución se utilizará el protocolo llamado **Common Address Redundancy Protocol (CARP)**.

Para ello:

1. Detenga todo el laboratorio utilizando el comando `kathara lclean`
2. Copie la configuración de nginx del balancer para que inicie automáticamente:

```
cd balancer/etc/nginx/  
cp sites-available/balancer.example sites-enabled/balancer.conf
```

Edite `balancer.conf` y reemplace las direcciones X.X.X.X por las propias de cada servidor.

3. Edite el archivo `lab.conf` del laboratorio `kathara-lab_nginx` en su equipo y agregue las siguientes líneas para crear un nuevo servidor llamado *backup*, que está conectado a las mismas redes que *balancer*:

```
backup[0]=B  
backup[1]=C
```

4. Reemplace **todo el contenido** del archivo `balancer.startup` por lo siguiente:

```
dpkg --ignore-depends perl-base -i /packages/*.deb  
service networking restart  
service apache2 stop  
service nginx start
```

5. En la carpeta del laboratorio, cree un archivo dentro de `balancer/etc/network/interfaces.d/` llamado `ucarp.conf` y deje el siguiente contenido:

```
# interfaz real conectada a ISP  
auto eth0  
iface eth0 inet static  
    address        203.0.113.201  
    netmask        255.255.255.0  
    gateway        203.0.113.1  
    ucarp-vid      1                # id del servidor virtual  
    ucarp-vip      203.0.113.200    # direccion IP virtual compartida  
    ucarp-password aygr2020        # clave para cifrar mensajes carp  
    ucarp-advbase  1                # intervalo entre mensajes de estado  
    ucarp-master   yes              # define si este host es master
```



```
# interfaz real conectada al pool de servidores
auto eth1
iface eth1 inet static
    address    10.1.1.201
    netmask    255.255.255.0

# interfaz virtual (compartida) montada sobre eth0
iface eth0:ucarp inet static
    address 203.0.113.200
    netmask 255.255.255.0
```

6. Cree una copia de la carpeta `balancer` y de su contenido con nombre `backup` . Luego copie el archivo `balancer.startup` como `backup.startup` .

```
cd ~/kathara/kathara-lab_nginx
cp -r balancer backup
cp balancer.startup backup.startup
```

7. Edite el archivo `backup.startup` y reemplace el octeto `.201` por `.202` en las primeras dos sentencias que comienzan con `address` para cambiar la dirección IP del equipo `backup` en la interfaz que corresponde a la conexión con el ISP. En el mismo archivo, reemplace la línea `ucarp-master yes` por `ucarp-master no` . La última instrucción `address 203.0.113.200 no` debe modificarse.
8. Inicie el laboratorio y espere hasta que todos los equipos estén en línea.
9. Inicie una captura de tráfico con `wireshark` o `tcpdump` sobre el enlace **B**.
10. Desde el equipo cliente, obtenga la página web de la empresa con el comando `curl` :  

```
curl http://www.empresa.com/
```

Vuelva a realizar la consulta varias veces. ¿Responde siempre el mismo servidor del pool?
11. Analice la captura en curso y determine qué dirección MAC corresponde a la dirección IP `203.0.113.200`
12. Detenga el servidor `balancer` ejecutando el comando `service networking stop` y repita la petición del punto 10. ¿Se perdió disponibilidad? ¿Qué puede observar en la captura? ¿A qué dirección MAC está asociada la dirección IP `203.0.113.200` ?
13. Ahora indique, ¿a qué capa del modelo OSI brinda redundancia el protocolo CARP? ¿Sirve para otros servicios además de HTTP?
14. ¿Qué diferencia existen a grandes rasgos entre los protocolos CARP, VRRP y HSRP?
15. DESAFÍO: Ante un evento que obliga a un nodo secundario a tomar el rol del maestro,
  - a. ¿qué sucede con las conexiones TCP en las que el maestro era el destinatario?
  - b. ¿hay alguna forma de resolverlo?
  - c. ¿qué sucedería con las conexiones TCP si el maestro fuera un router y los segmentos sólo estuvieran siendo ruteados a través de él?