Administración y Gestión de Redes
Lic. Sistemas de información

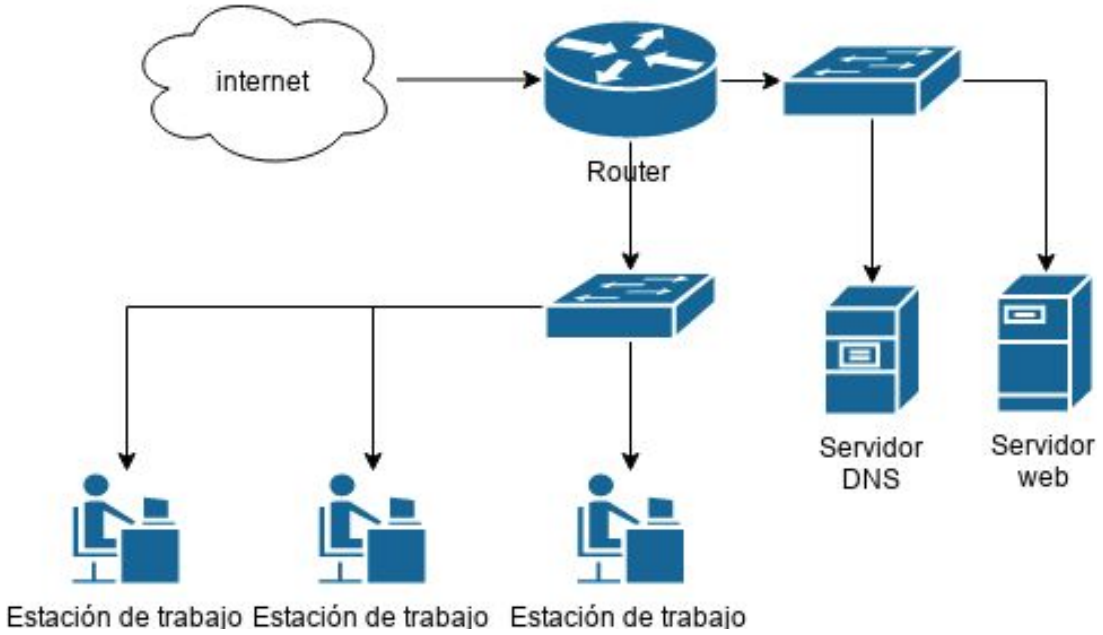
Laboratorio de Redes,
Recuperación de Información
y Estudios de la Web

Resiliencia y optimización

Alejandro Iglesias
aaiglesias@unlu.edu.ar
Mauro Meloni
maurom@unlu.edu.ar

v2019

Escenario de ejemplo



Jueguemos al juego...

¿Que tal si...

Conceptos asociados

Resiliencia (tolerancia a fallos): capacidad de un sistema para funcionar a pesar de fallos en algunos de sus elementos.

Alta disponibilidad (availability): aumentar el tiempo de disponibilidad de un servicio para sus usuarios autorizados.

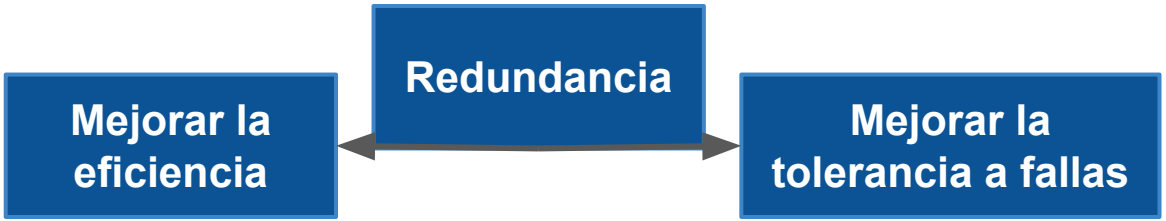
Distribución de carga (load sharing): distribuir entre dos o más elementos el trabajo necesario para brindar un servicio.

Balanceo de carga (load balance): distribuir de forma eficiente entre dos o más elementos el trabajo necesario para brindar un servicio.

Redundancia

La utilización de dos o más componentes en un sistema que son capaces de cumplir la misma función. El sistema debería ser capaz de seguir funcionando si se quita uno de estos componentes.

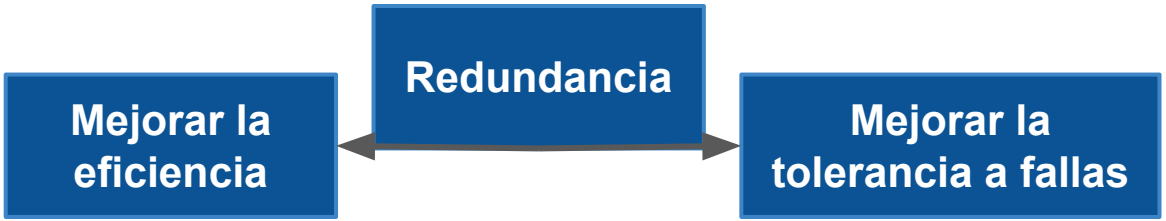
Moraleja: la redundancia sin una buena administración no aporta valor al sistema..



Redundancia

La utilización de dos o más componentes en un sistema que son capaces de cumplir la misma función. El sistema debería ser capaz de seguir funcionando si se quita uno de estos componentes.

Moraleja: la redundancia sin una buena administración no aporta valor al sistema..

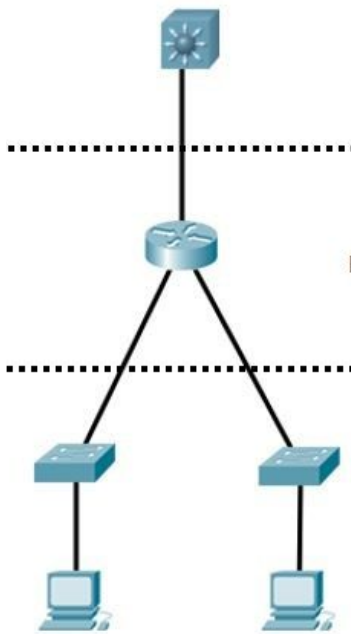


Redundancia en capas

En capas de TCP/IP

- Usuario
- Aplicación
- Transporte
- Red
- Enlace
- Física
- Infraestructura

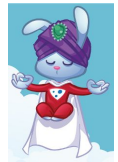
Modelo jerárquico



- Core
- Distribution
- Access

Redundancia en capas

Usuario	Múltiples administradores
Aplicación	DNS, DHCP, réplicas, HAProxy, Nginx. CDN
Transporte	Balanceo a nivel transporte. HAProxy, Nginx.
Red	Routers y hosts (BGP, VRRP, CARP, HSRP, Varnish)
Enlace	Enlaces lógicos (MSTP, Bonding,MPLS)
Física	Enlace físico, Hardware redundante (switches, servers)
Infraestructura	Energía eléctrica (UPS), control de temperatura (aires redundantes).



Patrones de diseño de redundancia
maestro/esclavo

Maestro + cold spare: el servicio corre en el maestro y el slave o spare está apagado.
Pros: Fácil configuración.
Contra: Tiempos de start up. Requiere acción humana.

Maestro + hot spare: el servicio corre en el maestro y el slave o spare está preparado y mantiene los datos sincronizados con la instancia principal.
Pros: No tan simple la configuración.
Contra: Requiere acción humana.

Maestro/esclavos: idem al anterior pero automático (heartbeat como mecanismo).
Pros: Automático
Contra: Mayor configuración y monitoreo.

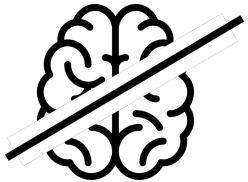
Patrones de diseño de redundancia
maestro/esclavo

Maestro/ slave read only: el servicio corre en el maestro y en el slave. Las escrituras en el master, y en ambos las lecturas.

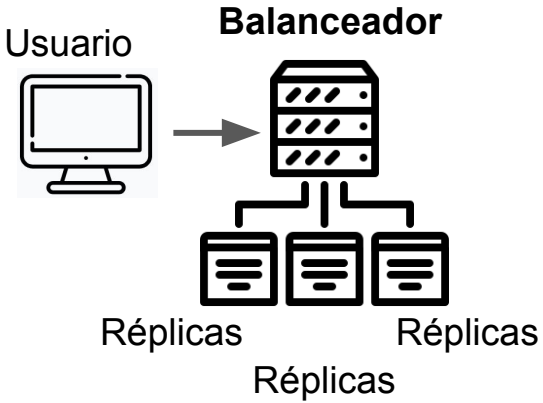
Pros: Compartición del trabajo de lectura.
Contra: Puede haber desfase entre master/slave.

Maestro/Maestro: ambos operan y realizan las mismas acciones.

Pros: Compartición del trabajo de lectura.
Contra: Métodos de sincronización. Posibilidad de Split Brain.



Patrones de diseño de redundancia
balanceador de carga/réplicas



Balanceador: recibe las peticiones al servicio y las reenvía según algún criterio a cada una de las réplicas.

Replicas: son capaces de recibir peticiones al servicio y resolverlas.

Ejemplo clásico: servidor web.

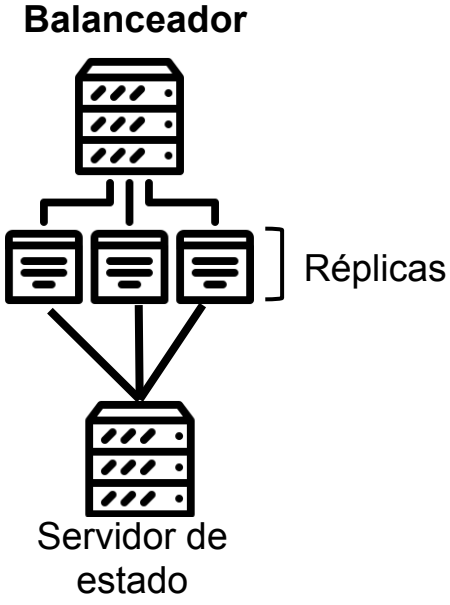
Pros:

- Mejor aprovechamiento de los recursos.
- Mejora en la eficiencia.
- Mayor escalabilidad.
- Mayor disponibilidad.
- Facilidad en la aplicación de actualizaciones/mantenimiento

Contra:

- Sincronización.
- Mayor número de equipos para proveer el mismo servicio.
- Qué pasa con los ”estados”

Patrones de diseño de redundancia
balanceador de carga/réplicas + state



Aquellos servicios que necesitan mantener un estado de las conexiones y sesiones de usuarios requieren de un elemento de sincronización entre las réplicas.

En el caso de los servicios web, por ejemplo, se necesita compartir la información de login en cookies entre las diferentes réplicas del servicio web.

Funciones de un balanceador

- Analizar las peticiones que recibe y decidir a qué servidor reenviarlas aplicando un algoritmo de planificación.
- Mantener un listado de servidores disponibles y su carga de trabajo.
- Proveer redundancia mediante el empleo de unidades múltiples ante un escenario de falla.
- Ofrecer distribución según el contenido, interpretando elementos de aplicación tales como URLs o cookies.

Estrategias para load sharing

- Round Robin (RR)
- Weighted RR
- Hashed
- Least Loaded (LL)
- Least Loaded with Slow Start
- Utilization Limit
- Least Time / Least Latency
- Power of Two Random Choices
- Cascade