Centros de datos

3º GREI – 1º Cuatrimestre

6 ETC´s — Obligatorios

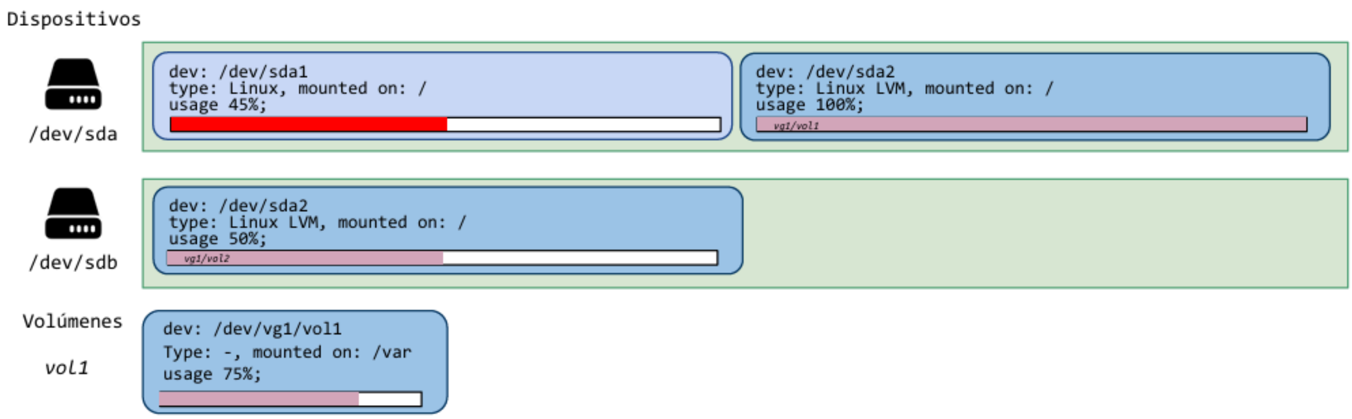
Almacenamiento en Centros de Datos

Def. Sistemas de almacenamiento:
Sistemas y dispositivos que permiten guardar y gestionar grandes volúmenes de datos y que aseguran que la información esté disponible de manera rápida y segura para usuarios y aplicaciones.

Utiliza tecnologías clave como volúmenes lógicos de almacenamiento y RAID.
Los sistemas de almacenamiento más usados son SAN y NAS.

LV

Los **volúmenes lógicos** son abstracciones de los volúmenes físicos para obtener mayor almacenamiento aprovechando mejor su espacio y permitiendo que varios dispositivos físicos integren el mismo volumen lógico como se aprecia en la siguiente imagen



Para gestionar estos volúmenes se emplea la siguiente herramienta, el **LVM** ((Logical Volume Manager).

LV Manager

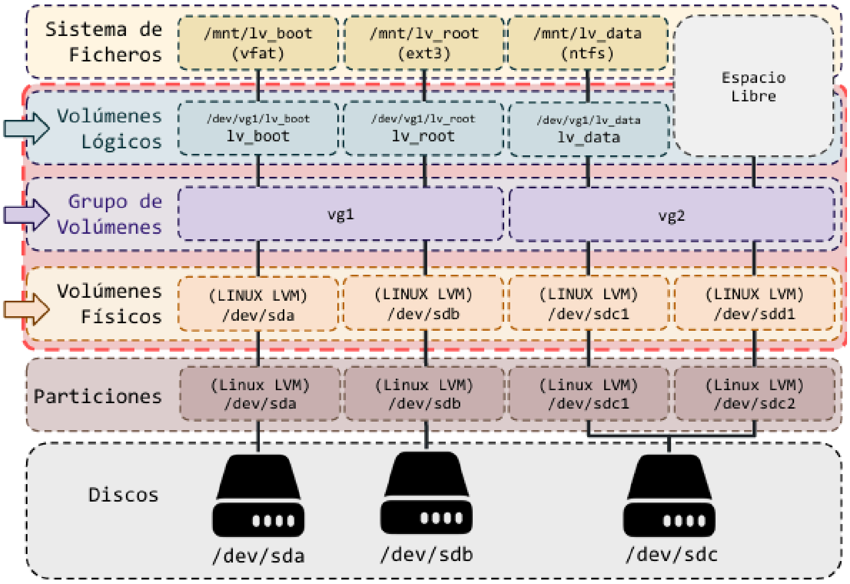
Como se menciona anteriormente esta herramienta ofrece la administración sobre la abstracción de los discos físicos a un formato lógico.
Esto tiene como siempre sus ventajas:
Ventajas

- Flexibilidad, administrar los volúmenes sin afectar al SO.
- Gestión, Agrupar y administrar de forma eficiente
- Escalabilidad
- Disponibilidad, permite RAID

Componentes clave para el LVM:

- Volumen físico (PV), fuente de almacenamiento física (HDD, SSD…) o lógico (Particiones)
- Grupos de volúmenes (VG)
- Volumen lógico (LV), creado a partir de un grupo de volúmenes

Funcionamiento



LV → Espacio lógico asignado dentro de un VG (~partición)

VG → Agrupación de PV´s

PV → Dispositivos de bloques sobre los que se organizan los volúmenes

Nota

Un **PV** pertenece a un único **VG**.
Un **VG** contiene uno o más **PV**.

Un **VG** puede contener múltiples **LV**.
Un **LV** se crea dentro de un único **VG**.

El LVM se maneja por
extensiones tanto físicas como lógicas,
Extensiones Físicas (PE), que conforman los PV, siendo la unidad más pequeña que estos pueden almacenar. En un VG todos los PV comparten el mismo tamaño de PE.
Extensiones Lógicas (PL), similares al PE siendo asociadas a estos mismo cuando los PV son asociados a un VG.

RAID

Def. RAID
Combina múltiples discos de almacenamiento en una sola unidad lógica (disco virtual) utilizado para mejorar rendimiento, capacidad y/o tolerancia a fallos.

Un raid en otras palabras lo que hace es permitir la combinación entre diferentes dispositivos de bloques y/o particiones.

Esto produce una serie de ventajas y desventajas:

Ventajas	Desventajas
Rendimiento, al dividir las tareas de RW Redundancia, al duplicar información Flexibilidad, permite gestionar los discos sin HW adicional. Rentabilidad	Consumo de recursos, requiere de usar la CPU y el SO para gestionar sus operaciones Compatibilidad limitada, no todos los SO soportan los RAID por software

Conceptos clave:

- **Data Stripping**, reparto de los bloques de datos entre varios discos
- **Mirroring**, réplica de los bloques de datos en distintos dispositivos
- **Paridad**, uso de bloques/segmentos con datos de paridad para la recuperación ante errores
- **Spare disks**, almacenamiento extra que reemplaza (hot spare o manual) a unidades fallidas (elemento adicional/complementario a los niveles estándar)

Los raid se implementan a nivel de software (GNU/Linux) o mediante hardware (Controladores o cabinas de discos) siendo estos de forma simple o anidados (RAID dentro de otro RAID)

Tipos de RAID

RAID 0 (Striping)

Los datos son distribuidos en bloques entre los dispositivos RAID 0(data striping), lo que mejora significativamente el rendimiento de lectura y escritura.

Ventajas: Alta velocidad de lectura y escritura, aprovecha toda la capacidad de los discos.

Desventajas: Sin tolerancia a fallos: un fallo de disco provoca pérdida total de datos.

RAID 1 (Mirroring)

Los datos se copian en discos mediante mirroring, lo que asegura la disponibilidad de los datos incluso si un disco falla. Sin embargo, la capacidad efectiva se reduce a la de un solo disco.

Ventajas: Alta disponibilidad y redundancia, fácil recuperación en caso de fallo.

Desventajas: Mal aprovechamiento del espacio (Tamaño Total/2), no mejora el rendimiento en escritura (aunque la lectura puede ser más rápida).

RAID 5 (Striping con Paridad Distribuida)

Este tipo de configuración requiere al menos 3 discos, con 2 destinados a datos y 1 para la paridad XOR. Los datos y la información de paridad se distribuyen cíclicamente entre los discos. La capacidad total es igual a (n-1) por la capacidad del disco más pequeño.

Ventajas: Equilibrio entre rendimiento, capacidad y redundancia, reducción del impacto de fallos (tolerancia a 1 disco).

Desventajas: Rendimiento de escritura menor debido al cálculo de paridad, la reconstrucción después de un fallo puede ser lenta.

RAID 6 (Striping con Doble Paridad)

RAID 6 es similar a RAID 5, pero incluye doble paridad para ofrecer una mayor tolerancia a fallos. Requiere al menos 4 discos (2 para datos y 2 para paridad XOR y NAND), y su capacidad total es igual a (n-2) veces la capacidad del disco más pequeño.

Ventajas: Mayor tolerancia a fallos (hasta 2 discos); buena opción para sistemas críticos.

Desventajas: Rendimiento de escritura más bajo que RAID 5 debido a cálculos adicionales de paridad; mayor tiempo de reconstrucción.

RAID 0+1 (Stripping+ Mirroring)

Combina striping (RAID 0) y mirroring (RAID 1). Requiere al menos 4 discos, distribuyendo los datos mediante striping (RAID 0) y luego realizando una copia mediante mirroring (RAID 1).

Ventajas: Buen rendimiento en lectura/escritura gracias al striping; más sencillo de implementar que RAID 10.

Desventajas: Menor tolerancia a fallos: si un conjunto RAID 0 completo falla, los datos se pierden; menor robustez que RAID 10.

RAID 10 (Mirroring + stripping)

Combina mirroring (RAID 1) y striping (RAID 0). Requiere al menos 4 discos, agrupando los discos en pares espejo (RAID 1) y luego distribuyendo los datos entre los pares (RAID 0).

Ventajas: Alto rendimiento en lectura y escritura; alta redundancia y tolerancia a fallos.

Desventajas: Muy costoso en términos de capacidad (50% de eficiencia).

RAID 50 (RAID 5 + RAID 0, también llamado RAID 5+0)

RAID 50 combina RAID 5 (paridad distribuida) con RAID 0 (striping). Requiere al menos 6 discos, distribuyendo los datos entre varios conjuntos RAID 5, que luego se organizan en RAID 0.

Ventajas: Buena combinación de redundancia, capacidad y rendimiento; mayor tolerancia a fallos que RAID 5 solo.

Desventajas: Complejo de configurar; impacto en rendimiento durante la reconstrucción.

Resumen Comparativo

RAID	MÍNIMO DE DISCOS	REDUNDANCIA	CAPACIDAD ÚTIL	RENDIMIENTO	TOLERANCIA A FALLOS
RAID 0	2	no	100%	Alto	N/A
RAID 1	2	si	50%	Moderado	1 Disco
RAID 5	3	si	((n-1)/n)	Bueno	1 Disco
RAID 6	4	si	((n-2)/n)	Moderado	2 Discos
RAID 0+1	4	si	50%	Alto	1 Disco/Espejo
RAID 10	4	si	50%	Muy alto	Múltiples (distinto espejo)
RAID 50	6	si	~(((n-1)/n)/2)	Mejor que RAID 5	Fallo de múltiples discos

	RAID 0	RAID 1	RAID 5	RAID 6	RAID 0+1	RAID 10	RAID 50
VENTAJA	Velocidad	Seguridad	Seguridad	Seguridad	Velocidad Seguridad	Velocidad Seguridad	Velocidad Seguridad
INCONVENIENTE	Seguridad	Coste	Coste	Coste++	Coste++	Coste++	Costes

Redes de almacenamiento

Conceptos

Primero hay que explicar los 3 tipos de conectores y formas de almacenamiento en red que hay:

INTERFAZ	CONECTORES PRINCIPALES	USOS COMUNES
SATA III	SATA de datos, SATA de energía, eSATA, mSATA	PCs, laptops, SSDs, HDDs domésticos
SAS-	SAS estándar, Mini-SAS (SFF-8087, SFF-8643), Dual-Port	Servidores, almacenamiento empresarial
Fiber Channel	LC, SC, ST, MTP/MPO, SFP/SFP+	Centros de datos, SAN

Direct Attached Storage (DAS)	**Network Attached Storage (NAS)**	**Storage Area Network (SAN)**
Acceso de datos a nivel de bloques. Dispositivos de almacenamiento conectados internamente. No requiere uso de redes de comunicación. Ideal para servidores individuales o estaciones de trabajo que necesitan almacenamiento rápido y dedicado. Conexión Cliente-servidor: SATA-III, SAS. Servidor-almacenamiento: SATA-III o SAS.	Acceso de datos a nivel de ficheros/directorios. Almacenamiento conectado a la red local. Permite el acceso de cualquier cliente autorizado. Requiere el uso de dispositivos de red. Ideal para compartir almacenamiento centralizado en hogares y empresas pequeñas. Conexión Cliente-NAS: Ethernet o Wi-Fi a través de la red local. SMB/CIFS (Windows). NFS (Linux/Unix). AFP (MacOS). NAS-Almacenamiento: SATA-III o SAS.	Acceso de datos a nivel de bloques Requiere el uso de dispositivos de red. Red de almacenamiento: conexión de los intermediarios con los dispositivos de almacenamiento. Red local: conexión de los clientes y servidores de aplicaciones con los intermediarios. Ideal para centros de datos con grandes volúmenes de datos y aplicaciones críticas. Conexión Cliente-Intermediario: Ethernet (iSCSI o Fibre Channel over Ethernet). Fibre Channel (FC) para conexiones de alta velocidad. NAS-Almacenamiento: Fibre Channel (FC). Ethernet (iSCSI, FCoE). InfiniBand

Estos 3 tipos de almacenamiento en red comparten una cosa en común y es su estructura en 3 capas basada en **Capa de almacenamiento**, **Capa de infraestructura** y **Capa de hosts**.

- **Capa de almacenamiento:**
    - Formado por los *tarjetas*: dispositivos de almacenamiento (SSD, HDD, ...) donde residen los datos.
    - Incluye las infraestructuras de los dispositivos de almacenamiento (racks, fuentes de alimentación, cachés internas, controladoras RAID (hardware), spare disks).
- **Capa de infraestructura:**
    - Contiene los dispositivos que dan soporte a la comunicación entre hosts finales y dispositivos de almacenamiento:
    - Contiene dispositivos y cableado específico de la red implantada (switches, routers, ...).

    - Permite multipath: acceso a discos por diferentes caminos (paths).
- Capa de hosts:
    - Formado por *initiators*: dispositivos que hacen uso del almacenamiento de bloques proporcionado por la SAN.
    - Incluye software adicional (multipath) y drivers.

Protocolos

Los principales protocolos son:

SCSI (Small Computer System Interface)

Es un protocolo avanzado que permite conectar múltiples dispositivos a la misma interfaz, ofreciendo alta velocidad y flexibilidad, usado principalmente en servidores y almacenamiento de alto rendimiento. SCSI es más complejo y costoso.

ATA (Advanced Technology Attachment)

También conocido como **IDE** en versiones más antiguas, conecta dispositivos de almacenamiento (como discos duros) a la placa base, es más sencillo y económico, y se usa principalmente en computadoras personales y servidores de bajo costo. **SATA** es la versión moderna de ATA, con mejor rendimiento y mayor velocidad.

FCP: Para *fibre channel*, alta velocidad, utilizado en SANs.
FCoE: *fibre channel sobre ethernet*, aprovecha la infraestructura Ethernet para SANs de alto rendimiento.
iSCSI: *SCSI sobre IP*, para acceso a almacenamiento remoto sobre redes IP (más accesible y económico que FCP).
AoE: *ATA over Ethernet*, para conectar dispositivos de almacenamiento ATA a través de Ethernet (menos común).
NVMe-oF: *NVMe sobre redes* de alto rendimiento, para almacenamiento ultrarrápido y baja latencia.

Sistemas de ficheros

DAS/SAN exponen sus datos a la red a nivel de bloque.
Sobre los dispositivos, los hosts crean sistemas de ficheros usando modelos convencionales (ext3, ext4, ntfs, fat32…), esto conlleva a una problemática a la hora de interactuar con ficheros varios hosts, esto se provoca debido a que los dispositivos asumen que solo un host accede a la vez, siendo concurrente no coordinado lo que puede llegar a provocar sobreescritura de datos y/o corrupción del fichero.

Por ello se opta por un sistema de ficheros del tipo ***cluster***;
Este sistema de ficheros coordina el acceso a los bloques de datos mediante el uso de bloques distribuidos.
- Distribución de datos → los datos se dividen y distribuyen entre los nodos
- Replicación de datos → Bloque se replica en otros nodos
- Estructuras simultáneas → Bloqueos de bloques para asegurar la consistencia
- Consenso → Algoritmos que coordinan la escritura y la integridad de los datos

Unos ejemplos de estos formatos son:

–OCFS2 (Oracle Cluster File System)
–GFS/GFS2 (Global File System 2)

Copias de seguridad

Def. Copias de seguridad
Replica exacta de la información resguardada almacenada en un formato estándar con un posible seguimiento de su ciclo de vida.

Las copias de seguridad son necesarias para la recuperación de información pérdida y para la continuidad planes de negocio, puesto que los datos son un activo muy valioso, a veces incluso regulado legalmente.

El concepto de copia de seguridad engloba tanto la información a salvar, como el modo de copia y recuperación.

En términos de copias de seguridad o Backup de ahora en adelante, se define el concepto de "Ventana de backup" que es el intervalo sobre el que se pueden realizar estos mismos sin entorpecer el funcionamiento del sistema, reduciendo en una alta disponibilidad de estos, y buscando mecanismos y estrategias más eficientes para su realización.

Otros conceptos a tener en cuenta en los backup son:

- Tiempo de creación → Que es el tiempo necesario, acotado por la ventana de backup
- Tiempo de almacenamiento → Que corresponde al tiempo que se va a ser almacenada está backup
- Tiempo de restauración → El tiempo que necesita para ser reconstruido el sistema

Todo esto esta ubicado en

- **On-line** → copia activa con acceso directo e inmediato (SAN, NAS)
- **Near-line** → copia inactiva, se puede poner activa automáticamente (MT con sistemas robotizados).
- **Off-line** → copia desconectada, se requiere intervención manual (SSD/HDD almacenados) Off-site → copias en ubicaciones separadas (Nubes de almacenamiento)
- Copias adicionales en ubicaciones externas al sistema (centros off-site)

A su vez ofrece protección ante fallos al replicar los backups en distintos medios para reducir las pérdidas de datos.

Una cosa a resaltar es que todo esto tiene un alto coste venido de, dispositivos, gestión de las copias, la ventana de backup, almacenamiento externo, etc.

¿Dónde se guarda todo esto?

- **Discos Duros (HDD):** Son utilizados para backups on-line u off-line. Ofrecen alta capacidad y acceso rápido, pero su principal desventaja es que pueden sufrir fallos mecánicos a largo plazo.
- **Cintas Magnéticas (Tape Drives):** Son ideales para backups a largo plazo. Tienen gran capacidad y son de bajo costo, aunque su acceso es secuencial, lo que puede hacerlos más lentos.
- **Unidades de Estado Sólido (SSD):** Son usadas para backups de acceso rápido. Ofrecen alta velocidad y resistencia, pero su coste es más alto en comparación con los discos duros.
- **NAS (Network Attached Storage):** Son adecuados para backups centralizados (**on-line/near-line**). Permiten acceso en red y ofrecen redundancia mediante RAID, aunque son vulnerables a fallos locales.
- **SAN (Storage Area Network):** Se usan para backups empresariales masivos. Ofrecen alta capacidad y rendimiento, pero su costo y complejidad son elevados.
- **Cloud Storage:** Son útiles para backups off-site. Son escalables y accesibles desde.

Estrategias de Backup

- **Datos a resguardar**: Dependen de la importancia de la información para la organización.
- **Frecuencia del backup**: Depende de la creación de datos (frecuencia con la que se generan), el coste de realizar las copias y las consecuencias de la pérdida de datos no resguardados. Esto puede variar desde backups diarios hasta semanales o mensuales.
- **Tiempo de almacenamiento de las copias**: Depende del dispositivo/soporte de almacenamiento utilizado. Algunos soportes pueden almacenar datos por períodos más largos, mientras que otros son más volátiles.

Tipos de Backup

- **Copia de seguridad completa (copia base)**:
  - Es una copia exacta de todos los datos en un momento determinado.
  - **Ventajas**: Restauración simple, ya que todos los datos están contenidos en una sola copia.
  - **Desventajas**: Requiere grandes cantidades de almacenamiento y puede ser costosa. Es la base para otras estrategias de backup.
- **Copia de seguridad incremental**:
  - Parte de una copia de seguridad completa. Solo incluye los cambios desde la última copia total.
  - **Ventajas**: Menos datos y tiempo de copia, ya que solo se respalda lo nuevo o lo modificado.
  - **Desventajas**: Necesita todas las copias incrementales para una restauración completa. Si falla una copia, se pierde la recuperación de los datos.
- **Copia de seguridad diferencial**:
  - Parte de una copia de seguridad completa. Solo incluye los cambios desde la última copia diferencial.
  - **Ventajas**: Solo se necesitan dos copias (copia base y última copia diferencial) para restaurar. Más rápido que una copia completa.
  - **Desventajas**: El tamaño de las copias incrementales puede aumentar con el tiempo, y puede convertirse en una copia base si se mantiene demasiado tiempo..

Procesamiento en Centros de Datos

Def. Clúster
Agrupaciones de nodos (junto con las infraestructuras de comunicación y almacenamiento asociadas) destinados a trabajar de forma conjunta para ofrecer un determinado servicio y/o realizar operaciones de cómputo específicas

Tipos de clúster (en referencia al balanceo de carga)S

Clusters de Balanceo de Carga (Load Balancing Clusters)

Los nodos se encargan de repartir la prestación de un servicio específico (como aplicaciones web, bases de datos, etc.).

- **Punto clave**: La repartición eficiente del trabajo entre los nodos.
- **Objetivo**: Atender el mayor número posible de peticiones del servicio, optimizando el rendimiento y la capacidad.

Clústeres de Alta Disponibilidad (Failover Clusters)

Los nodos aseguran la disponibilidad de un servicio incluso si ocurre un fallo o una caída en alguno de los sistemas.

- **Punto clave**: La tolerancia a fallos y la capacidad de recuperación automática (**failover**).
- **Objetivo**: Garantizar la continuidad y consistencia del servicio sin interrupciones significativas.

Clústeres de Alto Rendimiento (High Performance Clusters)

Los nodos trabajan conjuntamente para llevar a cabo tareas que requieren un cálculo intensivo, como el renderizado de gráficos, análisis de datos y predicciones.

- **Punto clave**: Procesamiento distribuido entre los nodos.
- **Objetivo**: Maximizar el rendimiento y la capacidad de cálculo, resolviendo tareas complejas de forma eficiente.

Como conceptos adicionales se añade la SNAT (Source-NAT) y el proxy

SNAT: NAT que traduce las direcciones privadas en IP públicas (mediante router) sin filtrado de paquetes, actúa en nivel 3.
Proxy: Intermediario entre dos equipos en distinta red que filtra paquetes, actúa en nivel 5.

Balanceo de Carga

Def. Balanceo de carga
Conjunto de equipos que comparten carga de trabajo y tráfico de red, compuesto por un back-end (nodos que ofrecen los servicios) y un front-end (servidores que distribuyen las solicitudes entre los nodos).

También tiene que tener escalabilidad tanto vertical (scale up), mejorar los nodos (añadir más recursos); como horizontal (scale out), agregar más servidores/nodos para mejorar las prestaciones.

Desventajas:
No ofrece alta disponibilidad:
- Punto único de fallo
- No gestiona redundancia (balanceador o infraestructura)
- No gestiona fallos en servidores back-end

Implementación del Balanceo de Carga

DNS
El funcionamiento de este balanceador requiere un servidor DNS para asignar nombres de dominio a servicios mediante una lista de IPs asociadas. Utiliza DNS Round Robin, que reordena la lista de IPs en cada resolución de nombres.

Ventajas: Bajo coste, ya que no requiere un servidor dedicado; escalable al actualizar registros DNS.

Inconvenientes: Balanceo no equitativo debido a las cachés DNS en los clientes; no detecta sobrecarga ni fallos en los nodos; punto de fallo si falla el balanceador (se recomienda usar redundancia).

Nivel 5 (IPROXY)

El balanceador toma decisiones de redirección basándose en el contenido de las cabeceras HTTP/HTTPS de las solicitudes.

Ventajas: Rendimiento mejorado, distribuye las solicitudes entre varios nodos; escalabilidad, permite añadir nodos según la demanda sin interrupciones; alta disponibilidad, redirige peticiones a nodos alternativos en caso de fallos.

Inconvenientes: Punto de fallo, si el balanceador falla, los nodos dejan de funcionar (se recomienda redundancia).

Ejemplos:

- HAProxy: haproxy.org
- Varnish Cache: varnish-cache.org
- NginX: nginx.com
- Apache HTTP Server: httpd.apache.org
- mod_proxy
- mod_proxy_balancer

Funcionamiento
Cliente envía solicitud al Balanceador
Balanceador retransmite solicitud Servidor
Servidor envía respuesta al Balanceador
Balanceador retransmite respuesta al Cliente

.

A nivel 3 (IP)

El balanceador redirige el tráfico basándose en las direcciones IP y los puertos de las solicitudes.

Funcionamiento: Distribuye las peticiones entre los nodos según las direcciones IP de origen de las solicitudes.

Linux Virtual Server (LVS):
Algoritmos: Ofrece diez algoritmos de balanceo diferentes.
Marco de balanceo de carga de código abierto incluido en el kernel de Linux, utilizando el framework NETFILTER.
Componentes:

- **IPVS (IP Virtual Server):** Toma decisiones de balanceo de carga a nivel IP.
- **KTPVS (Kernel TCP Virtual Server):** Maneja el balanceo de carga a nivel de transporte.

Ventajas: Alta disponibilidad; compatible con la mayoría de servicios TCP/UDP.

<table>
<tr><th colspan="3">Implementación</th></tr>
<tr><th>Virtual server via NAT</th><th>Virtual server via IP-Tunneling</th><th>Virtual server via Direct-Routing</th></tr>
<tr><td>Cliente envía solicitud al Balanceador</td><td>Cliente envía solicitud al Balanceador</td><td>Cliente envía solicitud al Balanceador</td></tr>
<tr><td>Balanceador retransmite solicitud al Servidor (uso D-NAT)</td><td>Balanceador encapsula (paquetes) y envía la solicitud al Servidor</td><td>Balanceador encapsula (tramas) y envía la solicitud al Servidor</td></tr>
<tr><td>Servidor envía respuesta al Balanceador</td><td rowspan="2">Servidor desencapsula (paquetes) y envía respuesta directamente al Cliente</td><td rowspan="2">Servidor desencapsula (tramas) y envía respuesta directamente al Cliente</td></tr>
<tr><td>Balanceador retransmite respuesta al Cliente (uso D-NAT)</td></tr>
</table>

Estrategias del Balanceo de Carga

- **Round-Robin**:
  Reparte las cargas entre los servidores de forma alterna, similar al "reparto de cartas".
  - **Asume**: Todos los servidores tienen la misma capacidad de procesamiento y las cargas de trabajo son similares.
- **Weighted Round-Robin**:
  Asigna un peso a los servidores según su capacidad de procesamiento.
  - **Reparte** las cargas proporcionalmente al peso de cada servidor.
- **Least Connection**:
  Asigna las cargas al servidor con menos conexiones abiertas, buscando distribuir las peticiones de manera más equilibrada según la carga actual de cada servidor.

- **Weighted Least Connection**:
  Envía las peticiones al servidor que tenga la mejor relación entre conexiones abiertas y el peso asignado.
  - Ejemplo:
    - Servidores con peso y conexiones abiertas:
      - Servidor 1: **6 conexiones**, **peso 5** → relación 6/5 = **1.2**
      - Servidor 2: **2 conexiones**, **peso 1** → relación 2/1 = **2**
    - El servidor con **mejor relación** (en este caso el Servidor 2, con relación 2) recibe la siguiente solicitud.
- **Asignación Estática**:
  Asigna de manera fija los servidores reales en función de las direcciones IP de origen, garantizando que ciertas IPs siempre se dirijan a los mismos servidores.

Persistencia de conexión

Determinadas aplicaciones requieren mantener información sobre las conexiones o peticiones previas (es decir, sesiones), ya que **HTTP/HTTPS** es un protocolo sin estado, lo que significa que no guarda información sobre conexiones anteriores.

Balanceo de Carga sin Persistencia

En un balance sin persistencia, cuando el cliente realiza una nueva solicitud, el balanceador puede asignar un servidor distinto al anterior, lo que podría generar problemas si la aplicación depende de mantener una sesión continua.

Manejo de Persistencia

Session Replication (Replicación de Sesiones)

- **Descripción:** Se replica la información de las sesiones en todos los nodos del clúster, lo que permite que cualquier servidor pueda manejar la solicitud del cliente sin perder el estado de la sesión.
- **Ventajas:** Sencillez de implementación, ya que no requiere modificaciones significativas en las aplicaciones.
- **Inconvenientes:** Gasto de recursos, tanto computacionales como de almacenamiento.

2. **Sticky Cookies**
   - **Nivel de Aplicación:** Utiliza cookies para mantener la persistencia de la sesión a nivel de aplicación.
   - **Funcionamiento:** El balanceador o servidor asigna un ID único de sesión mediante una cookie en la primera respuesta (usando el encabezado Set-Cookie). En solicitudes subsecuentes, el cliente incluye esta cookie, y el balanceador usa el valor de la cookie para identificar al servidor original y redirigir la solicitud allí.
   - **Ventajas:** No requiere replicación de sesión ni almacenamiento adicional en el clúster.

Alta disponibilidad en Centros de Datos

Def Alta disponibilidad en centros de datos
Conjunto de nodos que garantizan la disponibilidad de un servicio ante paradas imprevistas (fallos) y previstas (mantenimiento), con comunicación entre nodos del clúster, nodos redundantes, capacidad de detectar fallos hardware/software, mantener el servicio operativo migrando a otro nodo y garantizar la integridad de los datos.

Esto en resumidas cuenta vienen a significar en evitar los puntos únicos de fallo (SPoF)) → Garantizar tolerancia a fallos sin provocar inconsistencias de datos

¿Dónde se usa la alta disponibilidad?
En aplicaciones y/o servicios críticos para una organización, como pueden ser; Bases de datos, aplicaciones de comercio, sistemas de ficheros compartidos…

Fiabilidad:
Capacidad del sistema de funcionar de manera consistente y predecible a lo largo del tiempo, sin experimentar fallos inesperados o interrupciones.
Calculada a través del MTBF (Mean Time Between Failures).

Disponibilidad:
Capacidad del clúster para estar en funcionamiento y proporcionar servicios a los usuarios cuando se necesita, expresada como el porcentaje de tiempo en que el sistema está disponible.
Grados altos de disponibilidad requieren mecanismos que permitan la operación continua incluso ante fallos o paradas imprevistas (tolerancia a fallos/failover).

Técnicas:

- Redundancia: Duplicación de nodos y recursos para garantizar disponibilidad.
- Detección y recuperación automática de fallos: Conmutación tras fallo para que otro nodo asuma los servicios de un nodo caído, minimizando el tiempo de inactividad.
- Balanceo de carga: Distribución equilibrada del tráfico y las tareas entre los nodos.

Facilidad de mantenimiento:

Capacidad de gestionar, actualizar y mantener el clúster de manera eficiente, minimizando el impacto en la disponibilidad de los servicios.
Calculada a través del MTTR (Mean Time to Repair).

Métricas

Uptime: → Tiempo total en funcionamiento del sistema.
Downtime: → Tiempo total fuera de servicio.
TTR (Time To Repair): → Tiempo transcurrido hasta poner en marcha el sistema después de un fallo.
TTF (Time To Failure): → Tiempo transcurrido hasta que el sistema falla.
TBF (Time Between Failures): → Tiempo transcurrido entre un fallo y otro.

MTTF (Mean Time To Failure): → Tiempo promedio transcurrido hasta que el sistema falla por primera vez. $\sum_{0}^{N} UptimeDispositivos/N$Dispositivos

MTTR (Mean Time To Recovery): → Tiempo promedio requerido para reparar el sistema y devolverlo a operación. $\sum_{0}^{N} Downtime/NFallos$

MTBF (Mean Time Between Failures): → Tiempo promedio entre fallos consecutivos del sistema. $MTTF + MTTR$

Availability (Disponibilidad): → $(Uptime/(Uptime + Downtime)) * 100$

Principios

Redundancia hardware: Replicación de componentes físicos clave (servidores, discos duros, fuentes de alimentación y tarjetas de red) con el fin de evitar puntos únicos de fallo.
++Disponibilidad de sistemas críticos --Costes

Redundancia software: Replicación de aplicaciones o servicios en diferentes servidores o nodos, de modo que si uno falla, otro asume la carga de trabajo.
++Flexibilidad, Puede funcionar en el hardware existente --Compleja configuración y administración

Redundancia de datos: Replicar los datos en múltiples ubicaciones o dispositivos para garantizar la disponibilidad e integridad de los datos.
++Asegura la protección de datos --Costes de almacenamiento, Complejidad en la gestión de los datos

Administración/Gestión de la redundancia: Software específico para gestionar los componentes redundantes para:
Asegurar la continuidad y correcto funcionamiento en caso de parada/caída imprevista
Garantizar la consistencia e integridad de los datos.

TIpos de cluster

Activo ⇔ Pasivo
Los nodos activos manejan las solicitudes de los clientes.
Los nodos pasivos están en espera y listos para asumir el control si un nodo activo falla.

- Esquema sencillo.
- Configuración simple.
- Uso ineficiente de recursos: los nodos pasivos están sin usarse.
- Costes de hardware duplicados: el nodo pasivo debe ser idéntico al nodo activo.
- Complejidad de la administración: sincronización de datos entre nodo activo y pasivo.

Sharding: divide los datos en partes más pequeñas (shards) y los distribuye en múltiples servidores.

- Sharding horizontal: datos o cargas de trabajo se dividen en particiones horizontales (por filas o registros) y se distribuyen entre nodos diferentes en el clúster.
  - Escalabilidad: más shards → más nodos
  - Carga distribuida: repartir → reducir la carga de los nodos

    - Consultas lentas: abarcan múltiples shards (totales)
- Sharding vertical: datos o cargas de trabajo se dividen por columnas o funciones y se distribuyen en diferentes nodos del clúster
    - Optimización: más shards → más nodos
    - Seguridad: repartir → reducir la carga de los nodos
    - Consultas lentas: múltiples columnas de distintos shards
    - Escalabilidad: limitada por las columnas

Activo ⇔ Activo
Todos los nodos manejan las solicitudes de los clientes al mismo tiempo.
Cualquier nodo puede servir como respaldo ante fallos en los demás nodos.
Si un nodo falla, las solicitudes se redistribuyen automáticamente a los nodos restantes.

- Alto rendimiento: todos los nodos comparten la carga.
- Tolerancia a fallos: si un nodo falla, otros nodos continúan atendiendo las solicitudes de los clientes.
- Complejidad de configuración y administración: complicado administrar múltiples nodos y gestionar la distribución de la carga → requiere herramientas de administración específicas.
- Latencia adicional: dado que los nodos comparten la carga, puede haber una ligera latencia adicional al procesar solicitudes de clientes.

Combinable con balanceo de carga → repartir la carga entre nodos.

Consideraciones

El fallo de un nodo supone que otro nodo/s pasa a hacerse cargo del recurso

def failover
Capacidad de recuperarse automáticamente de un fallo en un nodo desplegando el recurso en otro nodo.

La recuperación automática de servicios/aplicaciones debe garantizar la integridad de los datos.

Heartbeat: Mecanismo de sondeo del estado de los nodos mediante comunicación periódica. Si un nodo activo no responde, un nodo de respaldo lo reemplaza. Se implementa con red dedicada.

- **Aislamiento del tráfico:** Evitar que el tráfico de red y heartbeat compitan.
- **Evitar falsos positivos:** Prevenir errores de nodos caídos por saturación.
- **Replicación de redes heartbeat:** Mejora la tolerancia a fallos.

Integridad y consistencia
En configuraciones Activo ⇔ Activo:
El problema más frecuente es cuando múltiples nodos acceden simultáneamente a los mismos datos, lo que genera un riesgo de corrupción de datos.
Soluciones:

- Mecanismos de control de concurrencia (bloqueos/versiones).
- Mecanismos de consenso distribuido.
- Transacciones: Agrupar operaciones para garantizar la atomicidad.

En recuperaciones ante fallos:
Split-brain: Ocurre cuando un nodo activo no responde a los heartbeats debido a un fallo de comunicación. El sistema asume que ha fallado y otro nodo toma su tarea, lo que provoca que ambos nodos realicen la misma tarea simultáneamente. A más nodos en el clúster, mayor riesgo de split-brain, lo que puede corromper los datos compartidos.

- Mecanismos de fencing (protección de acceso a datos compartidos):
    - El **quórum** es un mecanismo utilizado en clústeres para garantizar que solo la mayoría de los nodos activos controlen los recursos compartidos. Este sistema se basa en una comunicación constante entre los nodos y un sistema de votación, donde cada nodo tiene un voto. Para que los nodos puedan acceder a los recursos compartidos, deben alcanzar el quórum, es decir, más del 50% de los votos, asegurando así que no haya conflictos de acceso en caso de fallos o particiones en el clúster.
    -
    - **STONITH** es un mecanismo de seguridad que asegura que un nodo defectuoso no pueda acceder ni corromper los recursos compartidos en un clúster. Este sistema aísla los nodos fallidos mediante una monitorización constante, y en caso de detectar un fallo, obliga al apagado o reinicio del nodo afectado, ya sea mediante un proceso de hardware o software, para evitar que continúe afectando al funcionamiento del clúster.

Computación de altas prestaciones (Alto rendimiento)

Def. Clúster HPC (High-Performance Computing):
Un clúster HPC es un conjunto de computadoras interconectadas (nodos) que trabajan juntas como una unidad para resolver problemas de computación intensiva.

- Componentes clave:
    - **Nodos de cálculo**: Servidores que ejecutan partes del cálculo.
    - **Interconexión:** Redes de alta velocidad y baja latencia (InfiniBand).
    - **Almacenamiento compartido:** Almacenamiento centralizado para manejar grandes volúmenes de datos.
- Usos:
  Simulaciones físicas
  Bioinformática
  Predicción climática
  Inteligencia Artificial
  Análisis de datos masivos (Big Data)

Programación Paralela en HPC

Def. Programación Paralela

División de tareas computacionales en subtareas que se ejecutan en paralelo en diferentes nodos o procesadores.

Modelos de programación:

- **Memoria compartida:** Todos los procesadores acceden a la misma memoria.
  - **OpenMP**:
    - API para paralelizar código en sistemas de memoria compartida.
    - Aplicaciones:
      - Cálculos científicos complejos.
      - Simulaciones numéricas.
      - Aplicaciones que requieren control preciso sobre la ejecución de procesos.
- **Memoria distribuida:** Cada nodo tiene su propia memoria y se comunican entre sí.
  - **MPI (Message Passing Interface):**
    - Intercambio de mensajes entre nodos para coordinar las tareas.
    - Alto nivel, oculta detalles de software y hardware.
  - **MapReduce:**
    - Modelo para el procesamiento de grandes volúmenes de datos.
    - Divide el problema en tareas más pequeñas que se ejecutan en paralelo.
    - Operaciones básicas:
      - **Map:** Transforma el input en pares clave-valor.
      - **Shuffle & Sort:** Agrupa y ordena los pares por clave.
      - **Reduce:** Combina los valores para obtener el resultado final.

Hadoop

Def.Hadoop
Framework de código abierto utilizado para procesar grandes volúmenes de datos de manera distribuida en clústeres de computadoras diseñado para escalar de servidores individuales a miles de máquinas, cada una ofreciendo almacenamiento y procesamiento local.

- **Componentes clave:**
  - **Hadoop Distributed File System (HDFS):** Sistema de archivos distribuido que permite almacenar grandes cantidades de datos en múltiples nodos dentro de un clúster.
  - **MapReduce:** Modelo de programación utilizado para procesar grandes volúmenes de datos de manera paralela, dividiendo las tareas en pequeños fragmentos y ejecutándose simultáneamente en varios nodos del clúster.
  - **YARN (Yet Another Resource Negotiator):** Gestiona los recursos y la planificación de tareas dentro del clúster.

- Se usa en el procesamiento y análisis de grandes volúmenes de datos, análisis de Big Data en tiempo real, almacenamiento y procesamiento de datos no estructurados (logs, datos de sensores, etc.), aplicaciones en sectores como análisis financiero, biomedicina, telecomunicaciones.

- **Ventajas de Hadoop en HPC:**
    - **Escalabilidad:** Capaz de manejar petabytes de datos gracias a su diseño distribuido.
    - **Resiliencia:** Utiliza replicación de datos en diferentes nodos, lo que garantiza la disponibilidad en caso de fallos.
    - **Costo:** Hadoop puede funcionar en hardware estándar, lo que reduce los costos de infraestructura.
    - **Procesamiento paralelo:** Utiliza el modelo MapReduce para dividir tareas en subtareas que se ejecutan simultáneamente en diferentes nodos del clúster, acelerando el procesamiento.
- **Desventajas:**
    - **Latencia:** Aunque el procesamiento paralelo mejora la velocidad, la latencia en el acceso a datos distribuidos puede ser un desafío.
    - **Complejidad:** La gestión y mantenimiento de un clúster Hadoop puede ser compleja, especialmente en implementaciones grandes.
    - **Requiere gran cantidad de datos:** Hadoop es ideal para grandes volúmenes de datos; no es adecuado para tareas pequeñas o con baja cantidad de datos.

Redes y Seguridad en Centros de Datos

VPN	Acceso remoto e interconexión de redes
Firewall	Control de accesos y segmentación de redes
IDS/IPS	Monitorización de acceso y tráfico

VPN (Virtual Private Network)

Def. VPN
Conjuntos de tecnologías que extienden una red local(privada) sobre una red pública, manteniendo la confidencialidad del tráfico mediante un túnel virtual cifrado entre dispositivos y un servidor remoto.

Características:

- Oculta la dirección IP y ubicación.
- Evita bloqueos de sitios web.

Tipo VPN	Descripción	Configuración	Ejemplo	
PaP	Conecta redes locales entre sí, enfocada en la interconexión de redes completas	Más compleja: configurar dispositivos de red y establecer políticas de seguridad.	**Sede A:** 192.168.12.0	**Sede B:** 10.10.10.0

Acceso Remoto	Permite conectarse de forma segura a una red corporativa desde ubicaciones remotas, enfocada en usuarios o dispositivos individuales	Sencilla	**Red Casa:** 172.16.0.0	**Red Interna:** 10.10.10.0

Tecnologías

IPSec (Internet Protocol Security)

Def. IPSec
Conjunto de protocolos diseñados para proteger las comunicaciones de red a nivel de IP, proporcionando autenticación, integridad y, opcionalmente, confidencialidad.

Compatibilidad: Funciona sobre redes IPv4 e IPv6, compatible con protocolos TCP y UDP.

Autenticación y Protección:

  - **Autenticación (AH)**: Verifica la identidad de los participantes en la comunicación.
  - **Integridad (AH, ESP)**: Asegura que los datos no hayan sido alterados durante la transmisión.
  - **Confidencialidad (ESP)**: Cifra el contenido de los paquetes para proteger la privacidad de la información.
- **AH** (Authentication Headers): Proporciona integridad y autenticidad de los datos desde el origen utilizando funciones HMAC (Hash-based Message Authentication Code) con claves secretas.
- **ESP** (Encapsulating Security Payloads): Proporciona integridad y autenticidad (con HMAC) y confidencialidad (mediante cifrado simétrico).
- **ISAKMP** (Internet Security Association and Key Management Protocol): Para el intercambio de claves y para fijar/acordar los parámetros de las Asociaciones de Seguridad (SAs).

Este protocolo opera en 3 modos de protección, entre dos dispositivos (host-to-host), entre redes completas (network-to-network) o entre un dispositivo y una red (host-to-network).

Asociación de Seguridad (SA): Es un enlace lógico unidireccional que define los parámetros de seguridad (clave, algoritmo de hash, duración de las claves) para la comunicación, identificado por un SPI (Security Parameters Index) único.

Cifrado: IPSec permite el uso de algoritmos de cifrado como AES o 3DES para proteger el tráfico de red.

Autenticación y Confidencialidad: IPSec usa tecnologías como IKE (Internet Key Exchange) para el intercambio seguro de claves.

Modos de funcionamiento de IPSec:

- **Transporte**: Protege solo la carga útil (datos de la capa de transporte y superiores). Permite enrutamiento transparente, evitando la encriptación del encabezado IP. Es común en comunicaciones host-to-host.
- **Túnel**: Protege todo el paquete IP (cabecera y carga útil), encapsulando el paquete completo dentro de un paquete IPSec. Se utiliza habitualmente en comunicaciones network-to-network entre pasarelas IPSec.

WireGuard:

Protección de tráfico a nivel IP, usa solo UDP, compatible con IPv6.

Seguridad:

- **Autenticación e integridad**: Poly1305.
- **Confidencialidad**: ChaCha20.

Modos: Host-to-host, network-to-network, host-to-network.

Estado estático: Conexión VPN con claves públicas/privadas predefinidas.

Proceso de Handshaking:

- Generación de claves: Curve25519 (clave pública/privada).
- Semilla: Combinación de claves mediante ECDH.
- Clave simétrica (Ks): HKDF(Semilla) = Ks.
- Encriptación/Desencriptación: ChaCha20(Ks XOR Datos a cifrar/descifrar).

OpenVPN

OpenVPN es una familia de protocolos que protege el tráfico a nivel de Transporte/Aplicación. Es uno de los estándares más utilizados para establecer conexiones VPN seguras. Utiliza la biblioteca OpenSSL para crear enlaces cifrados extremo a extremo utilizando SSL/TLS, además de proporcionar soporte para la creación de certificados digitales mediante criptografía.

Modos de operación:

- **Host-to-Host**: Enlace cifrado entre dos máquinas.
- **Road Warrior**: Conexión remota a un servidor OpenVPN desde fuera de la red.
- **Red a Red**: Conexión entre dos redes separadas creando una red unificada, con tráfico cifrado.

Funcionamiento:

- OpenVPN establece una conexión SSL cifrada entre los dos extremos usando la red pública (Internet).
- Se crean interfaces de red virtuales (como tun0, tun1, etc.) en los equipos conectados. Estas interfaces funcionan de manera similar a las interfaces de red tradicionales, como eth0 y eth1, proporcionando funciones como asignación de IP, enrutamiento y filtrado.

- El tráfico IP que reciban y envíen estos equipos se encapsula dentro de la conexión SSL y se transmite de forma cifrada.

Métodos de autenticación y cifrado:

- **Autenticación mediante clave compartida (Pre-Shared Key, PSK)**:

    Es un método de autenticación sencillo donde ambos extremos comparten una contraseña para conectarse a la VPN, con la ventaja de una configuración sencilla y la desventaja de baja seguridad si no se usa adecuadamente.

- **Autenticación mediante SSL (certificados digitales)**:
    - **Generación de Certificados**: Se genera un par de certificados digitales (clave pública y privada) tanto para el servidor VPN como para los clientes. Estos certificados deben ser firmados por una Autoridad Certificadora (CA) reconocida por todos los participantes.
    - **Proceso de establecimiento de conexión**:
        1. El servidor muestra su certificado al cliente para que este pueda verificar que es legítimo.
        2. El cliente valida el certificado del servidor utilizando las claves públicas de la CA y del servidor.
        3. El servidor solicita al cliente que se autentique mediante su propio certificado.
        4. El servidor verifica la autenticidad del cliente usando el certificado de este último.
        5. Si ambos se autentican con éxito, se establece una conexión VPN segura.

Firewall

Def. Firewall
Barrera de seguridad que se utiliza para controlar y filtrar el tráfico de red entrante y saliente.

- **Hardware**: Incluye componentes físicos como la CPU, memoria, tarjetas de red y almacenamiento, necesarios para operar el firewall.
- **Software**: Se refiere a la aplicación firewall (como `iptables`, `shorewall`, etc.) que ejecuta las reglas de filtrado, junto con sistemas adicionales como IDS (Sistema de Detección de Intrusos), IPS (Sistema de Prevención de Intrusos), y servicios VPN.

Funciones del Firewall

Separación de Redes:
Un firewall protege y aísla una red interna (de confianza) de una red externa, que puede ser potencialmente hostil.

Prevención de Intrusiones:
Detecta actividades maliciosas en tiempo real y puede bloquear ataques como los intentos de acceso por fuerza bruta.
Los firewalls pueden bloquear direcciones IP que intentan acceder repetidamente a recursos de forma no autorizada.

Control del Uso de la Red:
Permite monitorizar y controlar qué tráfico es permitido o denegado en la red interna.

Decisión sobre los Paquetes:
Analiza los paquetes de datos que atraviesan el firewall y decide, con base en reglas definidas, si deben ser permitidos o bloqueados.

- Esto se hace en función de la dirección IP, el puerto, el protocolo y otros atributos del paquete.

Tipos de Firewalls

- **Filtros de Paquetes (Stateless Firewalls):**
  Estos firewalls analizan los paquetes individuales sin mantener el contexto de las conexiones. Evalúan cada paquete de manera aislada según reglas predefinidas de IPs, puertos y protocolos.

  Ventajas: Simplicidad y bajo coste.
  Desventajas: No tiene en cuenta el estado de la conexión, lo que puede hacerlo más vulnerable ante ataques sofisticados.

- **Filtros con Estado (Stateful Firewalls):**
  Mantienen un registro de las conexiones activas y examinan los paquetes dentro del contexto de estas conexiones. Analizan si un paquete pertenece a una conexión existente, lo que permite tomar decisiones más informadas.

  Ventajas: Mejor seguridad y eficiencia, ya que se basa en el contexto de la conexión.
  Desventajas: Mayor complejidad y uso de recursos, ya que necesita almacenar el estado de las conexiones.

- **Filtros a Nivel de Aplicación (Proxies):**
  Se encargan de controlar el tráfico a nivel de aplicación, analizando y filtrando las solicitudes y respuestas entre el cliente y el servidor.

  Ventajas: Permite un control más detallado del tráfico y puede ofrecer funciones como la priorización de aplicaciones o bloqueo de contenido no deseado.
  Desventajas: Requiere configuración más compleja y recursos adicionales.

  - **Proxy Directo (Forward Proxy):**
    Actúa como intermediario entre los usuarios internos y los servidores externos, ocultando las direcciones IP de los usuarios internos y bloqueando el acceso a ciertos sitios.

- **Proxy Inverso (Reverse Proxy):**
  Opera entre los clientes externos y los servidores internos, protegiendo las direcciones IP de los servidores internos, ayudando en balanceo de carga y mejorando el rendimiento y seguridad web.
- Además, puede optimizar la privacidad y seguridad de los servidores internos al gestionar las solicitudes antes de que lleguen a estos.

Topologías de Firewalls

Cortafuegos Básicos de Borde (Single Firewall):

- Se coloca un único equipo como firewall entre la red interna y externa.
- **Ventajas**: Simplicidad y coste bajo.
- **Desventajas**: Si este firewall se ve comprometido, toda la red interna está en riesgo.

Host Oculto (Screened Host):

- Un firewall se coloca entre la red interna y los servicios de la red externa, con una máquina "bastión" en el interior.
- **Ventajas**: Aísla los servicios internos de los externos, mejorando la seguridad.
- **Desventajas**: La máquina bastión puede ser vulnerable si no se configura adecuadamente.

Host Inseguro (Untrusted Host):

- Se coloca un equipo no seguro en la red externa. Este equipo ofrece servicios a los usuarios sin comprometer la red interna.
- **Ventajas**: Permite ofrecer servicios externos sin debilitar la seguridad interna.
- **Desventajas**: El mantenimiento y gestión del host inseguro son complicados.

Zona Desmilitarizada (DMZ):

- Es una red intermedia entre la red interna y la externa, que aísla los servicios expuestos a Internet, mejorando la seguridad general de la red interna.
- **Ventajas**: Mayor seguridad para los equipos internos y mejor disponibilidad de los servicios expuestos.

Zona Desmilitarizada con Doble Firewall (Screened Subnet):

- Se coloca una DMZ entre dos cortafuegos, uno controla el tráfico desde la red externa hacia la DMZ y otro controla el tráfico desde la DMZ hacia la red interna.
- **Ventajas**: Proporciona una seguridad adicional al contar con dos capas de defensa, haciendo mucho más difícil que un atacante acceda a la red interna.

Técnicas de NAT (Network Address Translation):

- **SNAT (Source Network Address Translation)**: Traduce las direcciones IP internas (privadas) a una IP pública cuando los dispositivos internos acceden a recursos externos, permitiendo que múltiples dispositivos internos compartan una única IP pública.

- **DNAT (Destination Network Address Translation)**: Permite redirigir el tráfico que llega a una IP pública a una dirección IP interna específica. Es útil para exponer servicios internos a la red externa, como servidores web.

IDS/IPS (Sistemas de Detección/Prevención de Intrusiones):

Def. IDS/IPS
Los Sistemas de Detección de Intrusiones (IDS) y los Sistemas de Prevención de Intrusiones (IPS) son tecnologías de seguridad que monitorean redes o sistemas con el objetivo de detectar y prevenir actividades o accesos no autorizados, así como identificar patrones de comportamiento malicioso.
mientras que los IPS no solo detectan los ataques, sino que también toman acciones activas para bloquearlos o prevenirlos.

IDS

Los IDS generan alertas y registran los eventos detectados, estos se identifican en dos tipos:

- NIDS (Detector de Intrusiones en Red): Monitorea el tráfico de red en busca de actividades sospechosas, se instala en puntos clave como switches o routers, y detecta amenazas como escaneo de puertos, DoS o accesos no autorizados.
  - Ejemplos de NIDS:
    - SNORT: Un popular sistema de detección de intrusiones de red, disponible en snort.org.
    - Suricata: Un IDS/IPS avanzado, disponible en oisf.net.
- HIDS (Detector de Intrusiones en Host): Monitorea el sistema local analizando archivos de registro, configuraciones y actividades específicas, detectando modificaciones en archivos, cambios de permisos y eventos anómalos, pero no monitoriza el tráfico de red fuera del dispositivo.
  - Ejemplos de HIDS:
    - OSSEC: Un sistema de monitoreo y detección de intrusiones, disponible en ossec.net.
    - SAGAN: Un sistema HIDS compatible con reglas de Snort, disponible en sagan.quadrantsec.com.

IPS

IPS detectan los ataques y también toman acciones activas para bloquearlos o prevenirlos, se identifican 3 tipos de IPS:

- **NIPS** (Detector de Intrusiones en Red): Monitorea y analiza el tráfico de red en tiempo real para detectar y prevenir ataques, bloqueando o descartando paquetes o acciones sospechosas para evitar que los ataques lleguen a los sistemas internos.

- **HIPS** (Sistema de Prevención de Intrusiones en Host): Monitorea y previene actividades sospechosas directamente en el sistema operativo del dispositivo, bloqueando de manera activa las amenazas detectadas, similar al HIDS pero con capacidad de intervención.
- **WIPS** (Sistema de Prevención de Intrusiones en Wireless): Diseñado para proteger redes inalámbricas (Wi-Fi) contra ataques específicos, identificando y previniendo amenazas como ataques de desautenticación o intercepción de comunicaciones inalámbricas.

Abstracción de Centros de Datos

Def. Abstracción
Simplificación: oculta la complejidad del hardware y presenta una vista simplificada a las aplicaciones Permite que las aplicaciones se ejecuten sin tener que conocer los detalles específicos del hardware subyacente. Por ej. Sistemas de ficheros, procesos o E/S

Conceptos previos a tener en cuenta:
- **Emulador**: Permite ejecutar programas de computadora o videojuegos en una plataforma (arquitectura hardware o sistema operativo) diferente de aquella para la cual fueron escritos originalmente.
- **Simulador**: Se basa en recrear/replicar el comportamiento exacto de un sistema específico con el fin de imitar una realidad.
- **Virtualización**: Partiendo de la abstracción para crear diferentes entornos independientes entre sí utilizando los recursos (Hardware) de forma compartida y segura.
- **Hipervisor**: Software intermediario entre los recursos físicos y las máquinas virtuales (MV), gestionando los recursos así mismo como la gestión propia de estas MV.
- **Host**: Equipo físico donde se alojan los recursos físicos (Hardware) que emplean las VM.
- **Guest**: SO y aplicaciones que se comportan de forma independiente en una MV gracias al hipervisor.

Tipos de Virtualización

Virtualización completa:
- Permite crear varias MV sobre un único servidor físico mediante el hipervisor que se encarga de gestionar los recursos de cada máquina, existen dos variantes:
  - Tipo 1 → El hipervisor sustituye al SO
    - Micronúcleo: Incluye las funcionalidades fundamentales y algunas añadidas en forma de módulos externos
    - Monolítico: Incorpora funcionalidades básicas y añadidas directamente
  - Tipo 2 (Hosted) → El hipervisor se instala sobre un SO
    - Traducción binaria; Ejecución en código máquina para realizar labores críticas y en tiempo real
    - Ejecución directa: Ejecuta directamente las instrucciones seguras del sistema operativo invitado en el anfitrión sin traducción.

Paravirtualización

- Gracias a la ayuda de una API el *guest* puede comunicarse con el SO sin ayuda el hipervisor pidiendo los recursos que necesita de forma directa

Virtualización asistida por Hardware

- El procesador incluye extensiones que facilitan la ejecución de máquinas virtuales, permitiendo al hipervisor acceder directamente a ciertos recursos físicos sin necesidad de emuladores/API mediante la habilitación de un *RING-1* con privilegios adicionales.

Virtualización a nivel de sistema operativo

- Compartiendo el kernel, aplicaciones aisladas (en contenedores) se ejecutan en forma simultánea teniendo cada una su propio *user spac*e, bibliotecas y aplicaciones propias
    - Como componentes necesarios para su ejecución destacan:
        - Chroot → Crea un entorno aislado cambiando el /root
        - Kernel → Núcleo del SO compartido por todos los contenedores
        - Namespace → Aislamiento de contenedores mejorando eficiencia, seguridad y portabilidad
        - Cgroups → Limitador de recursos de un contenedor

Docker

Def. Docker
Plataforma de virtualización a nivel de sistema operativo que permite empaquetar, distribuir y ejecutar aplicaciones dentro de contenedores.

Def. Contenedor (Docker)
Entornos mutables ligeros, portátiles y aislados que incluyen todo lo necesario para que una aplicación funcione, como el código, las dependencias, bibliotecas y configuraciones, garantizando que se ejecute de manera consistente en cualquier sistema que soporte Docker.

Def. Imagen (Docker)
Paquete inmutable y auto-contenido que incluye todo lo necesario para ejecutar una aplicación. Es una plantilla de solo lectura a partir de la cual se crean los contenedores Docker, sirviendo como la base para ejecutar instancias de aplicaciones de manera consistente en cualquier entorno.

- Una imagen está compuesta por capas donde las superiores modifican a las inferiores. Un contenedor está formado por una o mas imagenes

Def. Volumen (Docker)
Persistencia de datos generados y utilizados por contenedores más allá de su ciclo de vida. Facilita el intercambio de datos entre contenedores, es gestionado automáticamente por Docker fuera del sistema de archivos del contenedor, y se almacena en una ubicación específica del host, asegurando aislamiento y persistencia.

Def. Bind Mount (Docker)
Tipo de montaje de almacenamiento que conecta una carpeta o archivo en el sistema de archivos del host con una ubicación dentro del contenedor. A diferencia de los volúmenes, que están gestionados por Docker, los bind mounts utilizan directamente el sistema de archivos del host y no están aislados de él.

Def. Linking (Docker)
Mecanismo que permite la comunicación entre contenedores, proporcionando un "túnel" para que un contenedor pueda acceder a servicios, recursos e información específicos de otro contenedor. Esto facilita la conexión entre aplicaciones que corren en diferentes contenedores sin necesidad de especificar manualmente las direcciones IP de los contenedores.

Redes en Docker
Docker proporciona un sistema de redes que permite la comunicación entre contenedores, gestionando cómo estos interactúan entre sí y con el mundo exterior. Para garantizar la seguridad y el control sobre el tráfico, Docker establece reglas de filtrado que impiden el tráfico entre contenedores en redes distintas. Existen tres tipos principales de redes en Docker:

- Bridge:
  - Comunicación interna entre contenedores dentro del mismo host por nombre o IP.
  - Conectividad externa indirecta mediante NAT (traducción de direcciones de red).
- Host:
  - Los contenedores usan la red del host directamente, sin NAT.
  - Máxima velocidad y eficiencia de red, pero sin aislamiento.
- None:
  - El contenedor no tiene interfaz de red, por lo que está completamente aislado sin comunicación interna ni externa.

Def. Dockerfile
Archivo de texto que contiene un conjunto de instrucciones que Docker usa para automatizar la creación de una imagen de contenedor. Este archivo define todo lo necesario para construir la imagen, como el sistema operativo base, las dependencias, las aplicaciones y configuraciones necesarias mediante una sintaxis específica.

Def. Docker Compose
Herramienta que permite definir y gestionar aplicaciones multi contenedor usando un archivo YAML que facilita la configuración de servicios, redes y volúmenes, y automatiza el despliegue de contenedores, como en el caso de aplicaciones web con múltiples componentes.
Se ejecuta con un solo comando “`docker-compose up`”, con el que puedes construir las imágenes necesarias y levantar todos los servicios, simplificando la gestión de aplicaciones complejas.

Virtualización vs Docker

La virtualización crea máquinas virtuales completas con sistemas operativos independientes, lo que implica un mayor consumo de recursos y tiempos de arranque más largos. En cambio, docker abstrae las aplicaciones del sistema operativo, permitiendo un uso más eficiente de los recursos y una ejecución más rápida, ya que los contenedores comparten el kernel del host sin necesidad de incluir un SO completo.

Cloud Computing

Def. Cloud Computing
Modelo de computación que permite el acceso a recursos de TI bajo demanda a través de Internet, eliminando la necesidad de infraestructura física in situ.

- Ventajas:
    - **Escalabilidad:** Los recursos pueden aumentar o disminuir según las necesidades de la empresa.
    - **Reducción de costes:** Se elimina la necesidad de invertir en hardware, redes o almacenamiento. Además, los gastos operativos se reducen al mínimo (mantenimiento, energía y personal técnico), y se paga solo por lo que se utiliza.
    - **Facilidad de mantenimiento:** Las tareas de mantenimiento son gestionadas por los operadores de Cloud Computing.
    - **Accesibilidad global 24/7:** Permite acceder a datos y aplicaciones desde cualquier dispositivo con conexión a Internet, lo que facilita la colaboración entre equipos distribuidos geográficamente.
- Inconvenientes:
    - **Dependencia de la conectividad:** Para acceder a los servicios en la nube, es necesario tener una conexión estable a Internet. Las interrupciones de la red pueden afectar la disponibilidad de los servicios.
    - **Problemas de seguridad y privacidad:** Los datos están controlados y almacenados por el proveedor de servicios, lo que implica depender de sus mecanismos de seguridad y normativas de privacidad.
    - **Dependencia de los proveedores:** Los problemas que afecten al proveedor impactarán directamente al cliente, y puede ser complicado migrar a otro proveedor debido a la integración y compatibilidad de servicios.

Nube Pública:
Consiste en una infraestructura compartida con otros usuarios, lo que permite aprovechar recursos de forma escalable y flexible bajo un modelo de pago por uso. Ofrece una amplia variedad de servicios como almacenamiento, cómputo y bases de datos.

- Sus principales ventajas son la rápida implementación y la buena relación calidad-precio.

- Inconvenientes como un bajo o nulo control sobre la infraestructura, lo que puede generar problemas de seguridad para los datos sensibles.

Nube Privada:
La infraestructura está dedicada a una sola organización, lo que brinda un mayor control sobre la seguridad y el cumplimiento normativo. Existen dos tipos de nubes privadas: *on-premise*, gestionadas internamente, y *privadas gestionadas*, en las cuales un proveedor externo se encarga de la gestión.

- Ventajas destacan el mayor control sobre la infraestructura y la seguridad, así como la posibilidad de personalizar la solución según las necesidades específicas.
- Inconvenientes su coste es más alto y su gestión es más compleja.

Nube Híbrida:
Combinación de nubes públicas y privadas, lo que permite la migración de cargas de trabajo entre ambas y ofrece flexibilidad para elegir la mejor opción para cada carga.

- Ventajas: optimizar costes, mejorar la flexibilidad y cumplir con los requisitos regulatorios.
- Inconvenientes, su complejidad de gestión es elevada.

Modelos de servicio

Infraestructura como Servicio (IaaS)

Modelo que proporciona a los usuarios el acceso (alquiler) bajo demanda a recursos informáticos (servidores, almacenamiento, red)
Ejemplos:

- Gestionar instancias de máquinas virtuales: Amazon EC2, Google CE, Microsoft, AVM.
- Gestionar almacenamiento escalable: Amazon S3, Google CS, Microsoft ABS
- Gestionar redes privadas: Amazon VPC, Google VPC, Microsoft AVN

Basado en la provisión (alquiler) de infraestructuras físicas básicas bajo demanda (servidores, almacenamiento) mediante virtualización.

- El proveedor ofrece el Hardware y los usuarios (clientes) tienen permisos basicos como instalar SO y aplicaciones necesarias
- Máquinas Virtuales: actúan como servidor físico independiente
    - Uso de hipervisores: Tipo 1/Tipo 2
    - Ejemplo: Amazon EC2
- Sistemas de almacenamiento en la nube: forma escalable/duradera de almacenar datos
    - Ejemplo: Amazon S3
- Redes Virtuales (VPCs): permite crear redes privadas y seguras en la nube
    - Ejemplo: Amazon VPC

Plataforma como Servicio (PaaS)

Modelo que proporciona a los usuarios el acceso (alquiler) bajo demanda de servicios/herramientas necesarias para realizar una actividad
Ejemplos:

- Heroku: plataforma para desarrollo de código online

- AWS Elastic Beanstalk: plataforma para el desarrollo y despliegue de aplicaciones en el ecosistema Amazon (AWS, EC2, S3)
- Microsoft Azure Apps: plataforma para alojar sitios web en ecosistemas Microsoft

Basado en la provisión (alquiler) de entornos colaborativos de desarrollo y despliegue de aplicaciones

- El proveedor ofrece el SO y aplicaciones contratadas y los usuarios (clientes) tienen permisos básicos como la configuración limitada del SO, uso y/o configuración del entorno y manejo de ficheros
- Máquinas Virtuales: actúan como servidor físico independiente
    - Uso de hipervisores: Tipo 1/Tipo 2
- Plataformas de ejecución:
    - Contenedores: aíslan las aplicaciones y sus dependencias en entornos ejecutables portátiles.
    - Orquestadores: automatiza el despliegue, escalado y gestión de contenedores
- Servicios Gestionados: bases de datos/cachés (mejoran el rendimiento de las aplicaciones), mensajería, ...
    - Uso directo y limitado: usuario interactúa directamente con el servicio con permisos limitados.
    - Uso transparente: el PaaS se encarga de gestionar las modificaciones de los servicios.

Software como Servicio (SaaS)

Modelo de distribución de software en la nube en el que las aplicaciones se alojan y se gestionan en los servidores de un proveedor.
Ejemplos:

- Microsoft 365
- Google Workspace
- Dropbox

Basado en el uso de aplicaciones de software mediante licencia distribuidas generalmente de pago

El proveedor ofrece el sistema operativo junto con la aplicación o recurso contratado, garantizando el cifrado de datos y el cumplimiento de la normativa LOPD, mientras que el usuario tiene permisos limitados para usar y configurar el servicio. Una única instancia del software brinda el servicio a múltiples usuarios, optimizando la distribución de recursos, garantizando seguridad e isolando los datos, todo esto a través de internet sin necesidad de instalaciones.

> **Ventajas:** Accesibilidad desde cualquier lugar, reducción de costos operativos, escalabilidad y flexibilidad de recursos, actualizaciones automáticas del software.
> **Desventajas:** Dependencia de una conexión a Internet, pérdida de control sobre el software y los datos.

Seguridad en la nube

Amenazas:

- Ciberataques:

    - Acceso no autorizado para obtener datos sensibles.
    - Ataques de denegación de servicio (DDoS) que sobrecargan servidores.
    - Vulnerabilidades en APIs que pueden ser explotadas por atacantes.
- Riesgos específicos de la nube:
    - Fallos en el aislamiento de multitenancy que exponen datos de otros usuarios.
    - Pérdida o fuga de datos sensibles.

Dependencia de terceros para garantizar seguridad y cumplimiento.

Las medidas de seguridad en la nube incluyen el cifrado de datos en tránsito y en reposo con gestión segura de claves, el uso de autenticación multifactor (MFA) y asignación de roles y permisos para controlar accesos, así como la realización de auditorías frecuentes y la monitorización continua del uso y acceso a los datos y servicios.

Gestión de la Nube

Pago por uso (On-demand): Flexibilidad inmediata sin contrato, pero costoso a largo plazo.
Instancias reservadas: Recursos a largo plazo con descuentos significativos, pero requiere compromiso constante.
Por capacidad o tamaño: Pago según recursos asignados; requiere ajuste para evitar sobre/infrautilización.
Por volumen o uso escalonado: Más consumo, menos coste; económico para altos volúmenes, pero caro para bajos.
Modelo Spot: Instancias sobrantes a precios bajos, aunque con disponibilidad no garantizada.
Suscripción o tarifa fija: Tarifa mensual/anual fija, con costes predecibles, pero puede ser ineficiente si no se aprovecha al máximo.
Por uso de funciones (Serverless o FaaS): Pago por tiempo exacto de ejecución, ideal para demandas ocasionales, no para ejecución constante.

La gestión de la configuración en la nube implica garantizar configuraciones coherentes y seguras mediante herramientas como Infraestructura como Código (IaC). Esto permite el control de versiones, la automatización del despliegue y la gestión de configuraciones mediante archivos, facilitando la replicación, auditoría y mantenimiento del entorno, además de permitir cambios seguros y revertir configuraciones problemáticas para aumentar la estabilidad y reproducibilidad.

Recuperación de la información

Los planes de recuperación ante desastres definen los pasos para restaurar sistemas en caso de fallos catastróficos, incluyendo respaldo y replicación de datos críticos, así como procedimientos de recuperación. Implican backups regulares, replicación en múltiples regiones y la implementación de sistemas redundantes para garantizar alta disponibilidad. Además, se realizan pruebas de resiliencia y simulaciones de fallos para evaluar y mejorar la eficacia de los procesos de recuperación.

Automatización

Def. Automatización

Las herramientas de gestión de infraestructuras surgen para automatizar o semi automatizar la administración de infraestructuras y servicios en los Centros de Datos. Su objetivo principal es **minimizar errores manuales**, **facilitar la escalabilidad** para manejar grandes volúmenes de servidores, **acelerar el despliegue de servicios y actualizaciones**, y **simplificar tareas administrativas**.
Estas herramientas se basan en la virtualización, centralizan la información de configuración, identifican y documentan tareas, fomentan buenas prácticas como la modularidad y reutilización de configuraciones, el uso de sistemas de control de versiones y permiten mejorar procesos de despliegue continuo mediante testing.

- Orquestación de Sistemas
  - La coordinación de la configuración y el despliegue de múltiples componentes en un sistema es esencial, especialmente en entornos que emplean virtualización, contenerización o cloud computing.
- Aprovisionamiento de Sistemas
  - Conjunto de procesos necesarios para preparar un equipamiento o infraestructura para que pueda prestar el servicio esperado
- Infraestructura como Código (IaC)
  - Estrategia que permite describir la infraestructura mediante código procesable de forma automática, este código debe ser "fácilmente" entendible por el administrador como por un software de automatización
- DevOps
  - Proceso que integra el desarrollo del software (Dev) y las operaciones/administración del software (Ops) para agilizar el despliegue y mejora contínua de las aplicaciones.

Aproximación a la Automatización

Herramientas Declarativas:

- Definen el estado final deseado.
- Gestión del estado automática (la herramienta aplica las acciones necesarias para alcanzar el estado deseado).
- Menos control sobre el flujo de ejecución.
- Se enfocan en el resultado final, no en los pasos detallados.

Herramientas Procedurales:

- Definen pasos específicos y el orden para alcanzar la configuración deseada.
- Menos automatización en la gestión del estado (requiere mayor intervención manual).
- Mayor control sobre el flujo de ejecución.
- Se enfocan en detallar las tareas específicas para lograr el estado deseado.

PULL

El enfoque *Pull* en la administración de configuraciones implica que los equipos clientes solicitan activamente configuraciones o actualizaciones desde un servidor central en intervalos definidos. En este modelo, los equipos se comunican con el servidor para verificar si hay cambios pendientes que aplicar, utilizando agentes que se conectan al servidor, obtienen las configuraciones necesarias y las implementan en los equipos. Herramientas populares que implementan este enfoque son *Puppet* y *Chef*.

Ventajas**:**

- **Escalabilidad:** No es necesario que el servidor central gestione directamente cada conexión, lo que facilita el manejo de grandes cantidades de clientes.
- **Tolerancia a errores:** Si un cliente pierde la conexión, puede reconectarse y aplicar las actualizaciones en su próximo ciclo de consulta.

Inconvenientes:

- **Dependencia de Agentes:** Requiere la instalación de agentes en cada equipo administrado, lo que puede ser una limitación en entornos sin acceso para instalar dichos agentes.
- **Intervalos de Actualización:** Los cambios no se aplican de inmediato si el intervalo de actualización es largo, ya que depende del ciclo de conexión del cliente con el servidor.

Push

El enfoque *Push* en la administración de configuraciones se basa en un servidor central que envía activamente las configuraciones o cambios a los equipos administrados, gestionando la distribución para asegurar que cada dispositivo reciba la configuración en el momento adecuado. El servidor inicia la comunicación con los equipos, "empujando" las configuraciones, y utiliza agentes en los dispositivos administrados para conectarse al servidor, obtener las configuraciones necesarias y aplicar los cambios. Herramientas como *Ansible* y *Terraform* se utilizan para implementar este enfoque.

Ventajas:

- **Aplicación inmediata de cambios:** El servidor puede enviar las configuraciones en tiempo real, lo que permite cambios rápidos y actualizaciones constantes.

- **Reducción de dependencia de agentes (modos sin agentes):** En la modalidad **agentless**, no es necesario instalar agentes en los dispositivos administrados, lo que facilita el mantenimiento y simplifica la infraestructura.

Inconvenientes:

- **Escalabilidad limitada:** A medida que el número de dispositivos administrados aumenta, la carga en el servidor central puede crecer, lo que podría limitar la capacidad de escalado.
- **Carga en el servidor central:** La distribución de configuraciones a muchos dispositivos simultáneamente puede generar una alta carga en el servidor, afectando su rendimiento.

- Puppet

Es una herramienta de automatización procedural que utiliza un lenguaje específico de dominio (DSL) basado en Ruby para describir el estado de los recursos mediante manifestaciones. Estos manifiestos son independientes de la plataforma y permiten la herencia. Opera bajo una arquitectura cliente-servidor con agentes instalados en los equipos administrados, que consultan al servidor periódicamente para aplicar cambios. Además, asegura *idempotencia*, lo que significa que las tareas solo se ejecutan si el estado actual del nodo no coincide con la configuración deseada.

- Ansible

Ansible es una herramienta de automatización declarativa que describe el estado de los recursos mediante un lenguaje basado en YAML. Utiliza playbooks para definir tareas y su estado de forma independiente de la plataforma, permitiendo la reutilización de tareas a través de roles. Funciona con una metodología *push* en una arquitectura cliente-servidor *agentless*, donde el servidor Ansible inicia las tareas en los nodos gestionados utilizando SSH.

Elementos clave de Ansible:

- Nodos:
  - *Nodo de Control*: equipo que ejecuta los comandos Ansible y se comunica con los nodos gestionados a través de SSH.
  - *Nodo Gestionado*: equipos que reciben y ejecutan las instrucciones desde el nodo de control sin necesidad de tener Ansible instalado.
- Inventory: documento que enumera los nodos gestionados y sus detalles de conexión.
- Playbooks: documentos YAML que describen las tareas a realizar en los nodos gestionados, con instrucciones específicas (tasks) y roles reutilizables.
- Módulos: scripts en Python que ejecutan tareas específicas como instalar paquetes, gestionar archivos o configurar servicios.
- Variables: permiten definir valores dinámicos reutilizables en playbooks, roles y tareas.
- Handlers: tareas que se ejecutan sólo si una tarea previa notifica un cambio.
- Templates: archivos dinámicos en formato Jinja2 para generar configuraciones personalizadas basadas en variables.

Ansible asegura *idempotencia*, lo que significa que las tareas solo se ejecutan si el estado actual de un nodo no coincide con la configuración deseada.

- Terraform

Es una herramienta de automatización declarativa que utiliza el lenguaje HCL (HashiCorp Configuration Language) para definir y provisionar infraestructura en múltiples proveedores de la nube. Emplea *resources* para describir los componentes y *modules* para facilitar la modularidad. La herramienta gestiona los cambios en la infraestructura a través de un modelo de estado y garantiza *idempotencia*, ejecutando las tareas sólo si el estado actual no coincide con la configuración deseada.