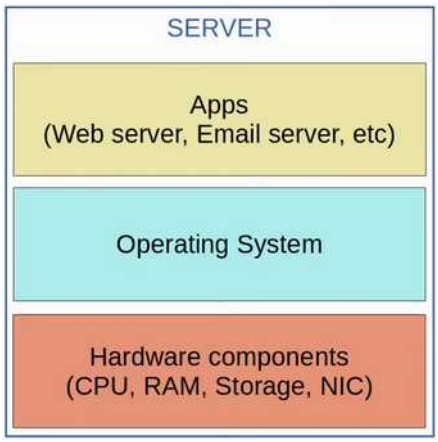
UD7-2 On premises, cloud y virtualización

Índice

Virtualización

Servidores antes de la virtualización



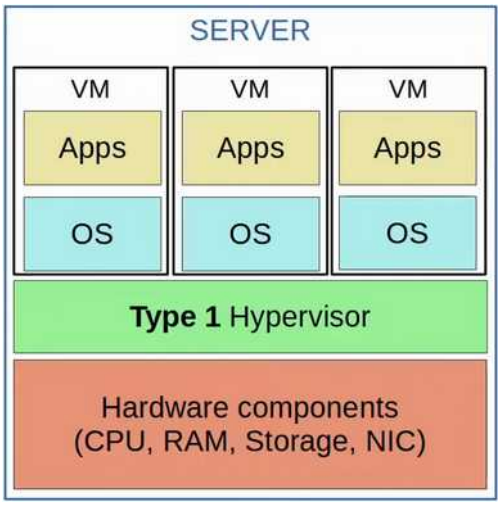
Antes de la virtualización, existía una relación uno a uno entre un servidor físico y un sistema operativo. En el diagrama puedes verlo: sobre los componentes de hardware del servidor hay un único sistema operativo en ejecución, por ejemplo, un servidor Linux. Dentro de ese sistema operativo, se ejecutaban aplicaciones que proporcionaban servicios como un servidor web, un servidor de correo, etc., como se muestra en el diagrama.

Se utilizaba un servidor físico para el servidor web, otro para el servidor de correo, otro para el servidor de base de datos, etc. Si bien era posible ejecutar todos estos servicios en un único sistema operativo en un solo servidor físico, esto no era recomendable, ya que, sin aislamiento entre ellos, un problema en una aplicación podía afectar a las demás. Sin embargo, usar un servidor físico por aplicación era ineficiente por varias razones:

1. **Costo y espacio**: Cada servidor físico es caro, ocupa espacio y consume energía.
2. **Subutilización de recursos**: La capacidad de CPU, RAM, almacenamiento y tarjetas de red (NIC, *Network Interface Card*, el componente que conecta el servidor a la red) solía estar infrautilizada.

Así funcionaban los servidores tradicionales.

Hipervisores de tipo 1

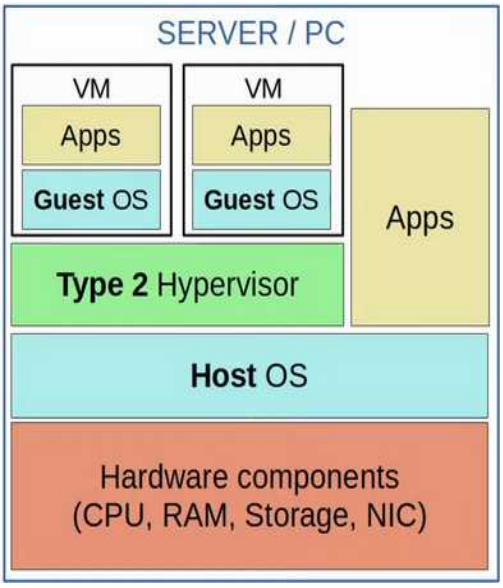


Con la virtualización, esto cambia. La virtualización rompe la relación uno a uno entre hardware y sistema operativo, permitiendo que múltiples sistemas operativos (SO) se ejecuten en un mismo servidor físico. Cada instancia se llama **máquina virtual (VM)**.

En el diagrama de la izquierda, tres máquinas virtuales se ejecutan en un solo servidor físico. Un **hipervisor** (o **VMM**, *Virtual Machine Monitor*) gestiona y asigna recursos como CPU y RAM a cada VM.

El Hipervisor Tipo 1: Se ejecuta directamente sobre el hardware (*bare-metal*). Ejemplos: VMware ESXi y Microsoft Hyper-V. Son eficientes porque consumen pocos recursos, dejando más para las VMs. También se les llama *hipervisores nativos* y son los usados en entornos de centros de datos.

Hipervisores tipo 2



Los **hipervisores Tipo 2** funcionan como un programa dentro de un sistema operativo (SO *host*). Ejemplos: VMware Workstation y Oracle VirtualBox. Puedes instalarlos en tu PC o portátil y ejecutar VMs dentro.

- **SO host**: El sistema operativo instalado directamente en el hardware.
- **SO guest**: El sistema operativo que se ejecuta dentro de una VM.

En el diagrama, un SO host ejecuta aplicaciones, incluido un hipervisor Tipo 2, y sobre este hay dos VMs con sus propios SO y aplicaciones. También se les llama *hipervisores alojados* (*hosted*).

Aunque rara vez se usan en centros de datos, son comunes en dispositivos personales (ej.: un usuario de Mac/Linux que necesita ejecutar una aplicación solo disponible en Windows).

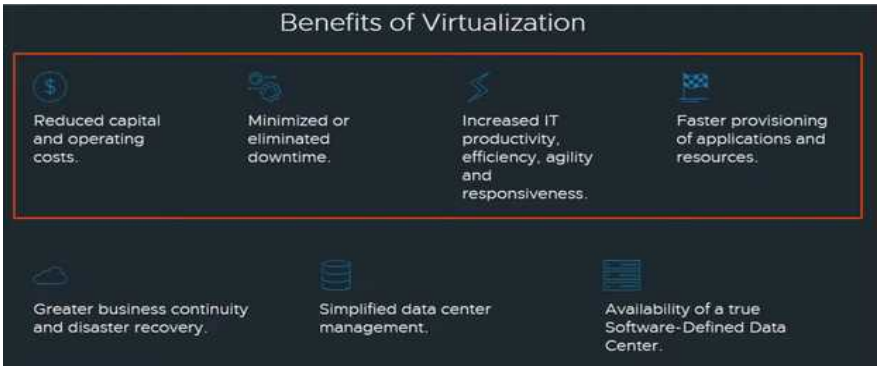
Ventajas de la virtualización

Según VMware, líder en virtualización, las ventajas clave son:

1. **Particionamiento**: Ejecutar múltiples SO en un mismo servidor, optimizando el uso de recursos.
2. **Aislamiento**: Fallos en una VM no afectan a las demás.
3. **Encapsulación**: Las VMs se guardan, mueven y copian como archivos normales.
4. **Independencia del hardware**: Las VMs pueden migrarse a cualquier servidor físico compatible con el hipervisor.

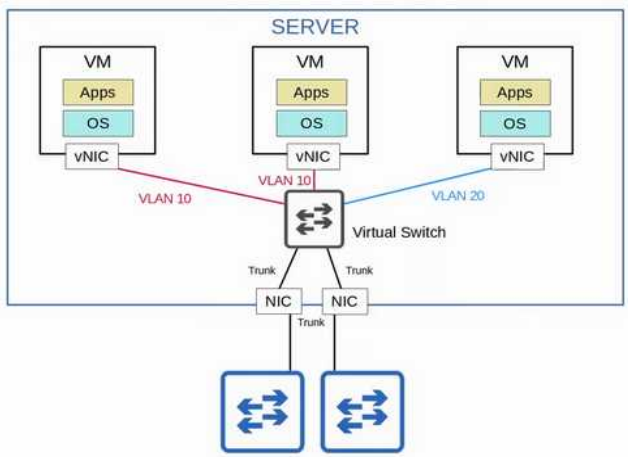
Beneficios adicionales:

- **Reducción de costos**: Menos servidores físicos = menos gastos en hardware, energía, espacio y refrigeración.
- **Menos tiempo de inactividad**: Facilidad para replicar VMs en servidores redundantes.
- **Mayor agilidad y velocidad**: Crear una VM es más rápido que comprar/configurar un servidor físico.



Redes virtuales

Las máquinas virtuales (VMs) pueden conectarse entre sí y a redes externas fuera del host físico mediante un **conmutador virtual (vSwitch)** que funciona en el hipervisor. Este vSwitch puede ser proporcionado por el hipervisor o ser un conmutador virtual de terceros, como Cisco.



Al igual que un conmutador físico, los puertos del vSwitch pueden configurarse como:

- **Puertos de acceso** (para una sola VLAN).
- **Puertos troncales** (para múltiples VLANs).

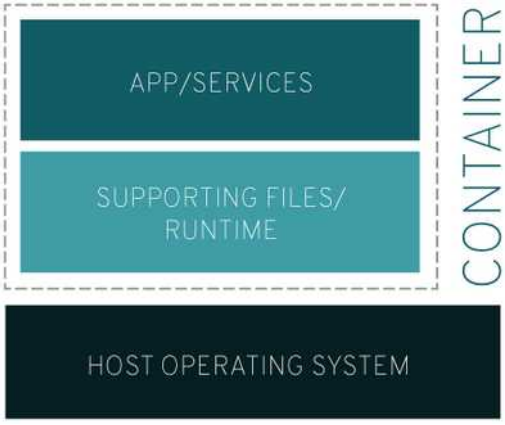
Por ejemplo, en el diagrama mostrado:

- Dos VMs están en la **VLAN 10**.
- Una VM está en la **VLAN 20**.

Las interfaces del vSwitch se conectan a las **tarjetas de red físicas (NICs)** del servidor para comunicarse con redes externas.

Contenedores

Podríamos definir un **contenedor** como *un paquete ligero, autónomo y ejecutable de una pieza de software, el cual incluye todo lo necesario para ser ejecutado: código, runtime, herramientas y librerías del sistema, así como sus configuraciones*.



Un **contenedor** es un proceso que se ejecuta de forma aislada, dentro de nuestro sistema operativo, al cual se le dota de las capacidades de ejecución básicas de un runtime. Estas capacidades nos permiten instanciar servicios y aplicaciones sobre el contenedor.

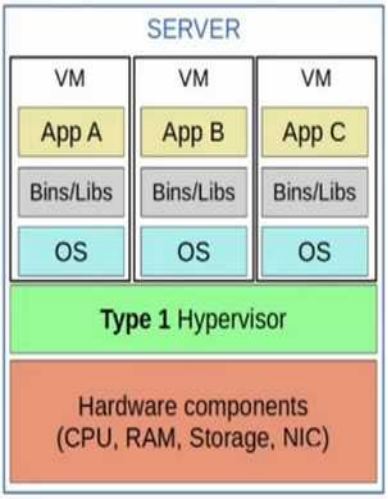
Esta configuración hace de los **contenedores** que sean muy ligeros y rapidos de arrancar a diferencia de otros modelos como podría ser las **máquinas virtuales**.

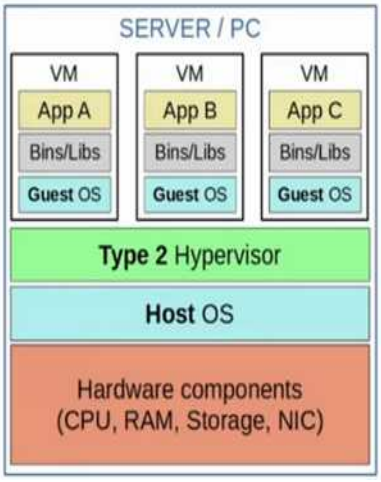
La idea es que dentro del contenedor podamos definir **los servicios básicos que necesita nuestro software**, las **configuraciones del software** y por último el **software en sí**. Todo queda autocontenido dentro del contenedor.

Al tener un elemento autocontenido, **el software se comporta siempre de la misma forma, independientemente del entorno en el que lo ejecutemos**. El software se aisla del entorno y de las diferencias que los entornos puedan tener en sus diferentes estadios: desarrollo, preproducción y producción, así como de las diferentes configuraciones que estos entornos puedan tener entre sí.

Esto nos permite reducir las configuraciones de los entornos, con el coste de tener igualados entornos de preproducción y producción, nos permite que el desarrollador inserte las configuraciones del software de una forma aislada y que los departamentos de explotación solo tengan que gestionar un elemento único independientemente de la tecnología que subyace por debajo.

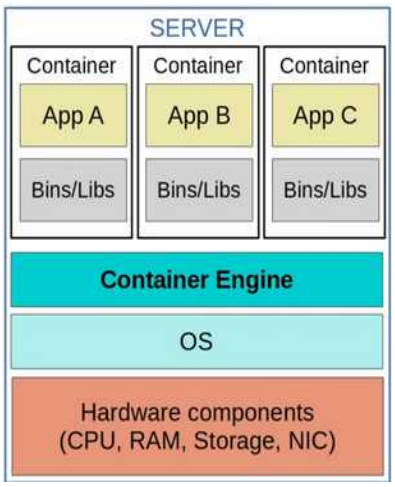
Contenedores





Las Máquinas Virtuales (VMs, por sus siglas en inglés) permiten que múltiples sistemas operativos se ejecuten en un solo servidor físico, y luego las aplicaciones se instalan en esos sistemas operativos. Se utiliza un hipervisor para gestionar y asignar recursos de hardware a cada VM.
Recuerda, todas las VMs se ejecutan sobre el mismo hardware físico, por lo que necesitan un hipervisor para asignar esos recursos adecuadamente.

Existen dos tipos principales de hipervisores: Hipervisores Tipo 1 se ejecutan directamente sobre el hardware y son ampliamente utilizados en centros de datos.
Hipervisores Tipo 2 que se ejecutan sobre un sistema operativo anfitrión, por ejemplo, Windows, macOS o algún tipo de Linux, se usan comunmente en dispositivos personales.
Sobre el hardware hay un sistema operativo anfitrión, con el hipervisor Tipo 2 y sus VMs instalados encima del sistema operativo anfitrión.
Por ejemplo, ejecutar un laboratorio de red virtual en tu PC usando Cisco Modeling Labs (CML).
Ten en cuenta que el sistema operativo en cada VM puede ser el mismo o diferente.
Si estás ejecutando múltiples VMs, una puede ejecutar Windows, otra Linux y otra macOS; no hay problema.
Los binarios y bibliotecas que he indicado en gris en los diagramas son bibliotecas de software y servicios necesarios para las aplicaciones que se ejecutan en cada VM.



Los contenedores son paquetes de software que contienen una aplicación y todas sus dependencias (los binarios y bibliotecas en el diagrama) para que la aplicación contenida pueda ejecutarse.

Ten en cuenta que se pueden ejecutar múltiples aplicaciones en un solo contenedor, aunque esta no es la forma habitual de usar los contenedores. En general, puedes asumir que un contenedor significa una aplicación.

Los contenedores se ejecutan en un motor de contenedores, como Docker Engine, que es el más popular. Y ese motor de contenedores se ejecuta en un sistema operativo anfitrión, generalmente Linux, que a su vez se ejecuta sobre el hardware.

Los contenedores son livianos e incluyen solo las dependencias necesarias para la aplicación específica. No es necesario ejecutar un SO en cada contenedor, a diferencia de las VMs.
Esa es la principal diferencia entre VMs y contenedores: las VMs ejecutan un sistema operativo en cada VM, mientras que los contenedores no. Todas las diferencias en costos y beneficios entre VMs y contenedores provienen de esa diferencia principal.

Un orquestador de contenedores es una plataforma de software para automatizar la implementación, gestión, escalado, etc., de contenedores. Por ejemplo, es posible que hayas oído hablar de Kubernetes, que es el orquestador de contenedores más popular. Docker también tiene uno llamado Docker Swarm.

¿Por qué necesitaríamos un orquestador de contenedores?
Bueno, en pequeñas cantidades, es posible operar contenedores manualmente, pero los sistemas a gran escala (por ejemplo, los que involucran microservicios) pueden requerir miles de contenedores. Esa cantidad de contenedores no se puede gestionar de manera realista de forma manual.

La Arquitectura de Microservicios es un enfoque de arquitectura de software que divide una solución más grande en partes más pequeñas, llamadas microservicios.
Así que, en lugar de una aplicación monolítica, podrías tener cientos de microservicios diferentes ejecutándose juntos para formar la solución más grande. Y esos microservicios se ejecutan en contenedores que pueden ser orquestados por Kubernetes u otra plataforma.

En primer lugar, las VMs pueden tardar minutos en arrancar, mientras que los contenedores pueden arrancar en milisegundos. Esto hace que los contenedores sean mucho más ágiles. Por ejemplo, si un contenedor falla, lleva muy poco tiempo que otro arranque y tome su lugar.

Además, las VMs ocupan más espacio en disco (a menudo decenas de gigabytes), mientras que los contenedores suelen ocupar solo decenas de megabytes. De nuevo, esto se debe a que cada VM incluye su propio sistema operativo, mientras que un contenedor solo contiene una aplicación y sus dependencias. Y por la misma razón, las VMs consumen más recursos de CPU y RAM en comparación con los contenedores.

En cuanto a la portabilidad, las VMs son bastante portátiles y pueden moverse entre sistemas físicos que ejecutan el mismo hipervisor, pero los contenedores lo son aún más. Son más pequeños, arrancan más rápido, y los contenedores de Docker pueden ejecutarse en casi cualquier servicio de contenedores.

Entonces, con estos puntos, los contenedores parecen mucho mejores que las VMs, pero las VMs sí tienen sus ventajas.
Una ventaja es que las VMs están más aisladas porque cada una ejecuta su propio sistema operativo. Un problema en un sistema operativo no afectará a las aplicaciones en otras VMs.
Sin embargo, eso no se puede decir de los contenedores: si el sistema operativo en el que se ejecutan todos los contenedores falla, todos los contenedores se ven afectados.
Ten en cuenta que este aislamiento también proporciona beneficios de seguridad para las VMs.

Cloud Computing

Tradicionalmente, las empresas implementaban su infraestructura IT de dos formas:

1. **On-premises (en las instalaciones)**:
    - Todos los servidores, routers, firewalls y otros dispositivos están ubicados en propiedad de la empresa.
    - La compañía es responsable de comprar, mantener y gestionar el hardware, así como del espacio, energía y refrigeración necesarios.
2. **Colocación (Colocation)**:
    - La empresa alquila espacio en un CPD para alojar servidores y equipos de red.
    - El centro de datos proporciona energía, refrigeración y seguridad física, pero la empresa sigue siendo dueña y responsable de sus dispositivos.

La Nube como Alternativa

Hoy en día, los **servicios en la nube** ofrecen una solución más flexible y escalable. Aunque muchos asocian la nube con proveedores públicos como **AWS**, existen diferentes modelos de implementación.

Definición de Computación en la Nube (según el NIST)

El **Instituto Nacional de Estándares y Tecnología de EE.UU. (NIST)** define la computación en la nube en su publicación **SP 800-145** como:

> *"Un modelo que permite el acceso ubicuo, conveniente y bajo demanda a un conjunto compartido de recursos informáticos configurables (redes, servidores, almacenamiento, aplicaciones y servicios) que pueden aprovisionarse y liberarse rápidamente con un esfuerzo de gestión mínimo o interacción con el proveedor de servicios."*

Este modelo se compone de:

1. **Cinco características esenciales**.
2. **Tres modelos de servicio**.
3. **Cuatro modelos de despliegue**.

Las cinco características esenciales de la nube

Para que un servicio sea considerado verdaderamente "en la nube", debe cumplir con **cinco características esenciales**:

1. **Autoservicio bajo demanda (On-demand self-service)**
    - *Definición del NIST*: *"El consumidor puede aprovisionar capacidades informáticas (como tiempo de servidor o almacenamiento) automáticamente, sin interacción humana con el proveedor."*
    - **Ejemplo**: En AWS, puedes crear o eliminar servidores virtuales desde un portal web
2. **Acceso amplio a la red (Broad network access)**
    - *Definición del NIST*: *"Los servicios están disponibles a través de redes estándar (como Internet) y son accesibles desde múltiples dispositivos (móviles, tablets, PCs, etc.)."*
    - **Ejemplo**: Puedes gestionar recursos en la nube desde un smartphone, una laptop o cualquier dispositivo conectado.

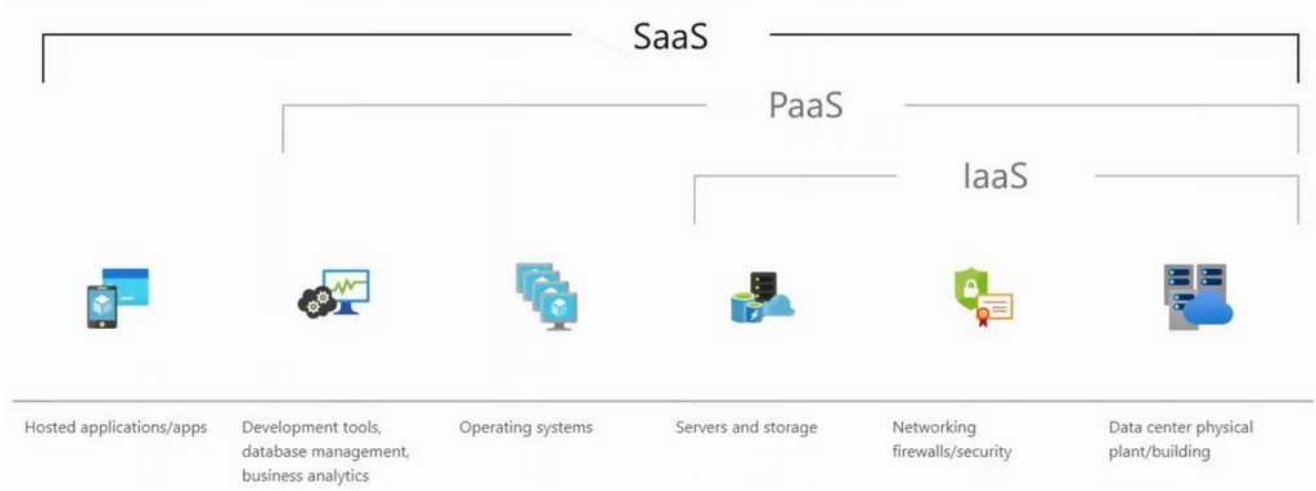
3. **Agrupación de recursos (Resource pooling)**
    - *Definición del NIST*: *"Los recursos del proveedor (almacenamiento, CPU, RAM, ancho de banda) se comparten entre múltiples clientes en un modelo multiinquilino. El cliente no controla la ubicación física exacta de los recursos, pero puede especificar regiones (país, centro de datos)."*
    - **Ejemplo**: AWS asigna dinámicamente recursos de su pool compartido cuando creas una VM.
4. **Flexibilidad rápida (Rapid elasticity)**
    - *Definición del NIST*: *"Los recursos pueden escalarse rápidamente (hacia arriba o abajo) según la demanda, dando la ilusión de capacidad infinita."*
    - **Ejemplo**: Si necesitas 100 servidores adicionales en minutos, AWS los provisiona sin negarte servicio.
5. **Servicio medido (Measured service)**
    - *Definición del NIST*: *"El uso de recursos se monitoriza, controla y reporta, permitiendo transparencia y facturación por consumo (ej.: GB de almacenamiento por día)."*
    - **Ejemplo**: AWS muestra tu consumo en tiempo real para evitar sorpresas en la factura.

Los tres modelos de servicio en la nube

En la nube, todo se ofrece "como servicio" (*as a Service*). Los tres modelos principales son:

1. **Software como Servicio (SaaS)**
   Definición del NIST: *"El cliente usa aplicaciones del proveedor (ej.: correo web) sin gestionar la infraestructura subyacente."*
    - **Ejemplos**: Office 365, Google Suite que incluye Gmail, Google Workspace.
    - **Responsabilidades**:
        - *Proveedor*: Gestiona todo (CPD, redes, servidores, SOs, aplicaciones).
        - *Cliente*: Solo usa la aplicación (ej.: Excel online).
2. **Plataforma como Servicio (PaaS)**
   Definición del NIST: *"El cliente despliega sus propias aplicaciones usando lenguajes de programación, librerías, servicios (como de BBDD) y herramientas del proveedor, sin gestionar infraestructura (redes, servidores, SOs)."*
    - **Ejemplos**: AWS Lambda, Google App Engine.
    - **Responsabilidades**:
        - *Proveedor*: Ofrece la plataforma (entorno de desarrollo, herramientas).
        - *Cliente*: Controla las aplicaciones que desarrolla.
3. **Infraestructura como Servicio (IaaS)**
   Definición del NIST: *"El cliente alquila recursos básicos (procesamiento, almacenamiento, redes) para ejecutar cualquier software, incluidos SOs y aplicaciones. El cliente controla los SO, el almacenamiento y las aplicaciones y posiblemente controle ciertos componentes de red como por ejemplo algunos firewalls"*
    - **Ejemplos**: Amazon EC2, Google Compute Engine.
    - **Responsabilidades**:
        - *Proveedor*: Mantiene la infraestructura física (centros de datos, hardware).
        - *Cliente*: Gestiona SOs, aplicaciones y configuraciones de red.

Fíjate que en SaaS Todo está controlado por el proveedor del servicio, En la PaaS Todo lo controla el Proveedor excepto las aplicaciones que son desplegadas por el cliente, relamente la PaaS es una plataforma que el Proveedor nos da para que programemos nuestras aplicaciones. Por último en la IaaS solamente la infraestructura el proveedor prové la infraestructura que hay por debajo.



Los cuatro modelos de implementación de la nube

Muchas personas asumen que la nube se refiere únicamente a proveedores de nube pública como AWS, Azure y GCP. Si bien es cierto que este es el modelo de implementación de nube más común, no es el único.

Aquí están los cuatro modelos de implementación de la computación en la nube: privada, comunitaria, pública e híbrida. Examinemos cada uno.

Nube privada

Según el NIST (Instituto Nacional de Estándares y Tecnología de EE. UU.):

"La infraestructura de la nube está provisionada para uso exclusivo de una sola organización, que puede incluir múltiples consumidores (por ejemplo, unidades de negocio). Puede ser propiedad de, administrada y operada por la organización, un tercero o una combinación de ellos, y puede existir dentro o fuera de las instalaciones de la organización."

Las nubes privadas generalmente solo son utilizadas por grandes empresas u organizaciones gubernamentales. Aunque la nube es privada (es decir, de uso exclusivo de una sola organización), puede ser propiedad de un tercero. Por ejemplo, AWS proporciona servicios de nube privada para el Departamento de Defensa de EE. UU. La infraestructura es de Amazon, pero está reservada para el uso exclusivo del DoD.

Las nubes privadas pueden estar dentro o fuera de las instalaciones (on-premises o off-premises). Esto es importante, ya que muchas personas asumen que la nube y las soluciones on-prem son cosas totalmente distintas, pero no siempre es así.

En una nube privada, se ofrecen los mismos tipos de servicios que en las nubes públicas: software, plataforma e infraestructura como servicio (SaaS, PaaS, IaaS), pero la infraestructura que conforma la nube está reservada para una sola organización.

Nube comunitaria

Según el NIST:

"La infraestructura de la nube está provisionada para uso exclusivo de una comunidad específica de consumidores de organizaciones que comparten inquietudes (por ejemplo, misión, requisitos de seguridad, políticas y consideraciones de cumplimiento). Puede ser propiedad de, administrada y operada por una o más de las organizaciones de la comunidad, un tercero o una combinación de ellos, y puede existir dentro o fuera de las instalaciones."

Este es el modelo de implementación de nube menos común. Es similar a la nube privada, pero la infraestructura está reservada para un grupo específico de organizaciones con intereses compartidos.

Nube pública

Ahora llegamos al modelo más popular: la nube pública. Según el NIST:

"La infraestructura de la nube está provisionada para uso abierto por el público en general. Puede ser propiedad de, administrada y operada por una empresa, una institución académica, una organización gubernamental o una combinación de ellas. Existe en las instalaciones del proveedor de la nube."

Este es, con mucho, el modelo de implementación más común. Algunos de los proveedores más populares de nube pública son: AWS, Azure, GCP, OCI, IBM y Alibaba, aunque AWS ha sido el líder indiscutible durante mucho tiempo.

Nube híbrida

Según el NIST:

"La infraestructura de la nube es una composición de dos o más infraestructuras de nube distintas (privada, comunitaria o pública) que permanecen como entidades únicas, pero están unidas mediante tecnología estandarizada o propietaria que permite la portabilidad de datos y aplicaciones (por ejemplo, cloud bursting para balancear cargas entre nubes)."

En otras palabras, este modelo es básicamente cualquier combinación de los tres tipos anteriores, en lugar de ser un tipo de implementación completamente nuevo. Un ejemplo sería una nube privada que puede descargar trabajo en una nube pública cuando sea necesario debido a limitaciones de recursos.