



UNIDAD FORMATIVA 3

Visualización cloud y contenedores

Visualización y cloud

Índice

Virtualización, cloud e IaC	2
Objetivos	4
Virtualización	4
Cloud computing	14
¿Qué es AWS?	18
Infraestructura como código	25
Referencias bibliográficas	31

Virtualización, cloud e IaC

DevOps se originó en las así llamadas empresas ‘nacidas en la web’ (empresas que se originaron en internet) como Etsy, Flickr y Netflix. Mientras estas empresas daban solución a los desafíos tecnológicos complejos de gran escala, tenían arquitecturas bastante simples, a diferencia de las grandes empresas que crecieron en torno a los sistemas heredados o a través de adquisiciones y fusiones, con sistemas de múltiples y complejas tecnologías que tenían que interactuar.

Como podréis imaginar, en la actualidad, todas las empresas de tecnología se enfrentan a nuevos retos y desafíos, como los modelos de entrega de aplicaciones móviles, las de nube y las cadenas de suministro de software.

En los siguientes apartados, explicaremos algunos de estos retos y, sobre todo, veremos cómo puede ayudar DevOps a resolverlos con la ayuda del cloud computing y la virtualización e introduciremos el concepto de infraestructura como código (IaC por sus siglas en inglés) y el impacto que tiene en el mundo tecnológico.

Veremos qué es la virtualización de hardware y la paravirtualización, ambos tipos basados en el concepto del **hipervisor** y cómo habilita una nueva utilización de los recursos físicos que tengamos a nuestra disposición en la organización: servidores, datacenters...

La virtualización es una tecnología extremadamente útil en varios escenarios, desde casos simples a extremadamente complejos. Lo que resulta importante es comprender cómo se desea utilizar la virtualización, ya que esto determinará qué solución de virtualización es la más adecuada para cada circunstancia u organización. Para ello, vamos a repasar algunos de los principales usos y casos de éxito de la virtualización.

Uno de los cambios más importantes que ha afectado a DevOps (y de hecho, también le ha ayudado a madurar) es el cloud computing (computación en la nube). La omnipresencia de la tecnología de nube privada, pública e híbrida ha permitido a las organizaciones proveer entornos bajo demanda. En las organizaciones tradicionales donde utilizan servidores físicos, solicitar un nuevo servidor es un proceso complejo, largo y tedioso.

Puede tomar días o incluso semanas adquirir un nuevo servidor físico, instalar y configurar el sistema operativo (OS) y configurar todo el software necesario para desplegar la aplicación. La virtualización y la nube ayudan a resolver ese problema en gran medida y este hecho ha traído aparejado un enorme crecimiento en la escala en la que se utilizan los entornos para desplegar las aplicaciones. A raíz de este fenómeno, se ha hecho necesaria la automatización de las tareas de implementación de software, abordadas y resueltas por DevOps.

Al mismo tiempo, la llegada de la nube facilita el despliegue continuo y proporciona un acceso más fácil a los entornos similares a producción durante las etapas de desarrollo y pruebas del canal de entrega. También ha llevado a un novedoso modelo de autoservicio para los desarrolladores quienes normalmente trabajaban de forma separada en relación con operaciones.

Con las tecnologías de nube, un desarrollador puede no solo realizar una construcción automática, sino también pedir el aprovisionamiento y configuración de un entorno de producción similar a la nube para desplegar la aplicación que está construyendo... ¡Todo ello sin la intervención directa del equipo de operaciones! Ambas capacidades proporcionadas por la nube facilitan la adopción DevOps.

Con el uso de clouds o las tecnologías de virtualización podemos definir, de forma precisa, el entorno de ejecución de nuestro software. Podemos también disponibilizar servicios añadidos, como pueden ser:

- Almacenamiento elástico.
- Bases de datos.
- Servicios de logs.
- Redundancia.

Llegados a este punto, nos volvemos a preguntar: ¿Se puede controlar toda esta necesidad de infraestructura de un modo organizado? ¿Cómo replicaríamos una puesta en marcha en otra cuenta, en otro entorno o con otros parámetros?

Hasta hace unos años los responsables de sistemas tenían que montar sus infraestructuras de hardware y software de forma manual. Es decir que montaban y apilaban el hardware, para posteriormente instalar y configurar los sistemas operativos y aplicaciones necesarias para el negocio. Como podréis imaginar, este proceso, además de ser muy tedioso, normalmente se ejecutaba por más de una persona, lo que generaba una larga espera hasta que la infraestructura de aplicaciones y sistemas estuviese disponible.

Para agilizar estas implementaciones, optimizar el rendimiento y entregar con más rapidez el software desplegado en las plataformas, los responsables de sistemas crearon scripts para unificar, automatizar y optimizar el proceso.

A su vez, la aparición de las metodologías ágiles ha instaurado una nueva forma de gestionar el software, que implica que la instalación, configuración y mantenimiento de los sistemas informáticos se haga más compleja, si cabe. Estas metodologías requieren de varios ambientes donde ejecutar las aplicaciones (desarrollo, testing, staging, etc.), lo cual termina duplicando, al menos, el esfuerzo de los administradores.

La respuesta que ha dado la informática a estos problemas es la optimización y automatización del despliegue de aplicaciones y la configuración de los servidores. Actualmente, es posible configurar servidores a través de la programación y sin intervención de los administradores. Esta nueva forma de administración, que considera a la infraestructura como un aplicativo más a ser gestionado de manera análoga al software, se las conoce como infraestructura programable o **infraestructura como código**, más conocida como **IaC** (*Huttermann, M., 2012*).

La infraestructura como código trata la infraestructura de configuración de sistemas como un software de programación. Esto genera una delgada línea entre los límites de la escritura de aplicaciones y la creación de entornos en los que se ejecutan. Se trata de una **parte fundamental** de la computación en la nube y esencial para DevOps. La IaC es el marco que ha dado origen a DevOps.

Objetivos

- Conocer las diferentes tecnologías existentes referentes a la virtualización.
- Conocer las diferentes herramientas que existen y los diferentes elementos susceptibles de ser virtualizados.
- Conocer los tipos de cloud existentes y los niveles de servicio que ofrecen.
- Explorar qué son los Amazon Web Services y cómo propone resolver los problemas a los que se enfrentan las organizaciones.
- Veremos qué es la infraestructura como código y qué opciones tenemos para implementarlo en un proyecto.
- Aprenderemos como dos de los principales actores, Terraform y AWS CloudFormation, describen un mismo recurso cloud como código.

Virtualización

Un paso más allá a la hora de utilizar los recursos físicos es el de la virtualización: aprovechar los recursos disponibles en un ordenador, o en una red de ordenadores, para generar 'máquinas virtuales' que los aislen y generen un entorno de ejecución o procesamiento.

La virtualización tiene varios usos comunes y lo cierto es que todos ellos giran en torno al concepto de que su tecnología representa una abstracción de los recursos físicos. De hecho, existen demasiados tipos de virtualización y es por ello por lo que resulta un tanto confuso tratar de determinar cuál es la mejor solución que se puede aplicar en cada organización.

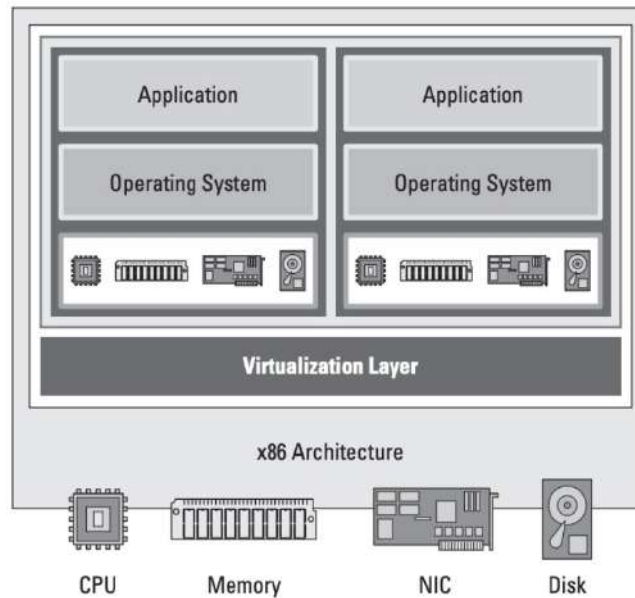
Dentro de cada uno de estos tipos existen, además, diferentes enfoques, cada uno con sus beneficios y desventajas: vamos a explorar estos 'sabores' básicos de las tecnologías de virtualización para que puedas descubrir las particularidades de cada uno y tomes decisiones estratégicas en la organización donde desarrolles tu actividad profesional.

Repasemos los dos tipos de virtualización basada en hipervisores:

1. Imitación o emulación del hardware

En este caso, el software de virtualización (**hipervisor**) crea una máquina virtual que imita todo el entorno de hardware.

El sistema operativo que está cargado en una máquina virtual es un producto estándar no modificado. Cuando realiza llamadas para recursos del sistema, el software de emulación de hardware captura la llamada del sistema y la redirige para que pueda gestionar estructuras de datos proporcionadas por el hipervisor. Es el propio hipervisor el que realiza las llamadas al hardware físico real, subyacente a toda la aglomeración de software.

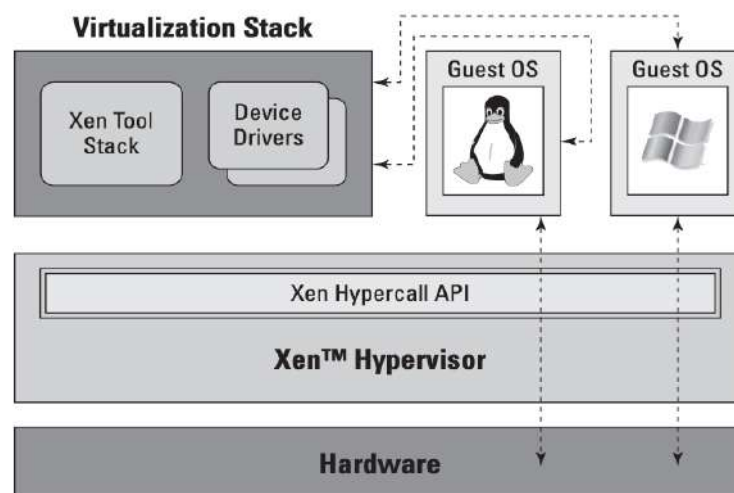


Virtualización por emulación de hardware. Fuente: Introducción a la virtualización por HP.

La emulación o imitación de hardware también es conocida como **virtualización de metal desnudo** (del inglés, *bare metal virtualization*), para simbolizar el hecho de que ningún software se encuentra entre el hipervisor y el 'metal' del servidor. Como hemos mencionado, el hipervisor intercepta las llamadas del sistema desde la máquina virtual huésped y coordina el acceso al hardware subyacente directamente.

2. Paravirtualización

La paravirtualización no intenta emular un entorno de hardware en software, sino que un hipervisor de paravirtualización coordina (o multiplexa) el acceso a los recursos de hardware subyacentes del servidor. La arquitectura de la paravirtualización se puede ver en la figura 2 que representa a Xen.



Estructura de la paravirtualización. Fuente: Introducción a la virtualización por HP.

En la paravirtualización, el hipervisor reside en el hardware y, por lo tanto, esta se puede concebir como una arquitectura de virtualización de metal desnudo.

Uno o más sistemas operativos huésped (equivalente a máquinas virtuales en virtualización de emulación del hardware) se ejecutan sobre el hipervisor. Un huésped privilegiado se ejecuta como una máquina virtual huésped, pero tiene privilegios que le permiten acceder directamente a ciertos recursos en el hardware subyacente.

Proveedores de virtualización

VMware, Citrix y Microsoft son los principales proveedores de virtualización de servidores x86 en entornos profesionales, más adelante, veremos otras opciones susceptibles de ser usadas también en entornos personales y de prueba.

VMware

Es el proveedor de virtualización de servidores más extendido y afianzado en el mercado. La plataforma insignia de VMware, vSphere, utiliza la tecnología de emulación de hardware.

Citrix

Ofrece un producto de virtualización de servidor llamado XenServer basado en paravirtualización. El huésped privilegiado (llamado control domain en lenguaje Xen) y el hipervisor Xen trabajan en equipo para permitir que las máquinas virtuales huésped interactúen con el hardware subyacente.

Microsoft

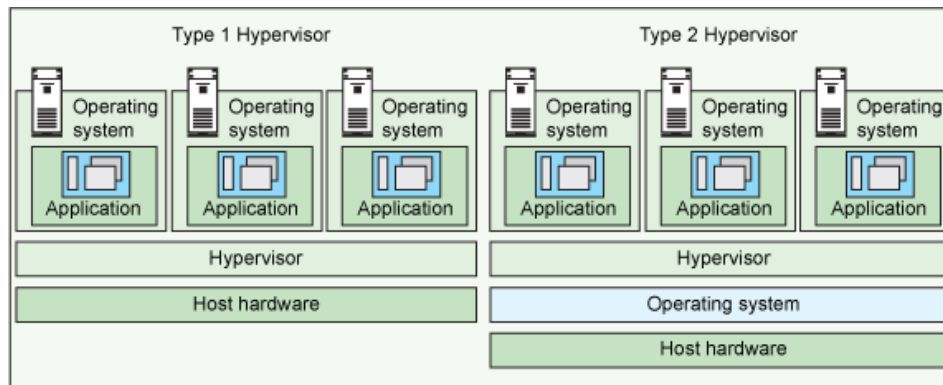
Hyper-V, el producto de virtualización del servidor de Microsoft tiene una arquitectura muy similar a la de Xen. En lugar de usar el término control domain para referirse a las máquinas virtuales huésped, Hyper-V se refiere a ellas como particiones y a la contraparte del control domain de Xen se la denomina partición principal.

Tipos de hipervisores

Una definición sencilla de hipervisor podría ser: la parte de la nube privada que gestiona las máquinas virtuales, es decir, es la parte (programa) que permite que múltiples sistemas operativos compartan el mismo hardware. Cada sistema operativo podría usar todo el hardware (procesador, memoria) si no hay otro sistema operativo encendido. Ese es el hardware máximo disponible para un sistema operativo en la nube.

Sin embargo, el hipervisor es el que controla y asigna qué parte de los recursos de hardware debe obtener cada sistema operativo, para que cada uno obtenga lo que necesita y no se interrumpa entre sí. Hay dos tipos de hipervisores:

- **Hipervisor de tipo 1:** Los hipervisores se ejecutan directamente en el hardware del sistema: un hipervisor integrado 'básico'.
- **Hipervisor tipo 2:** Los hipervisores se ejecutan en un sistema operativo host que proporciona servicios de virtualización, como soporte de dispositivos de E / S y administración de memoria.



Diferencias entre los tipos de hipervisores. Fuente: <http://www.ibm.com>.

Hipervisores tipo 1

- VMware ESX y ESXi

Estos hipervisores ofrecen funciones avanzadas y escalabilidad, pero requieren licencia, por lo que los costos son más altos. VMware ofrece algunos paquetes de menor costo y pueden hacer que la tecnología de hipervisor sea más asequible para infraestructuras pequeñas. VMware es el líder en los hipervisores tipo 1. Su producto vSphere / ESXi está disponible en una edición gratuita y 5 ediciones comerciales.

- Microsoft Hyper-V

El hipervisor de Microsoft, Hyper-V, no ofrece muchas de las funciones avanzadas que ofrecen los productos de VMware. Sin embargo, con XenServer y vSphere, Hyper-V es uno de los 3 principales hipervisores tipo 1.

Se lanzó por primera vez con Windows Server, pero ahora Hyper-V se ha mejorado enormemente con Windows Server 2012 Hyper-V. Hyper-V está disponible tanto en una edición gratuita (sin GUI y sin derechos de virtualización) como en 4 ediciones comerciales: Fundamentos (solo OEM), essentials, standard y datacenter. Hyper-V. Actualmente, las nuevas versiones de supervisores de Microsoft están íntimamente relacionadas a sus productos de cloud y se les conoce como 'Azure Stack'.

- Citrix XenServer

Comenzó como un proyecto de código abierto. La tecnología principal del hipervisor es gratuita, pero al igual que ESXi gratuito de VMware, casi no tiene características avanzadas. Xen es un hipervisor de tipo desnudo de tipo 1 y así como Red Hat Enterprise Virtualization usa KVM, Citrix usa Xen en el XenServer comercial.

Hoy, los proyectos y la comunidad de código abierto de Xen están en Xen.org. Hoy, XenServer es una solución comercial de hipervisor tipo 1 de Citrix, que se ofrece en 4 ediciones. Confusamente, Citrix también ha calificado sus otras soluciones propietarias como XenApp y XenDesktop con el nombre Xen.

- Oracle VM

El hipervisor Oracle se basa en el código abierto Xen. Sin embargo, si necesita soporte de hipervisor y actualizaciones de productos, le costará. Oracle VM carece de muchas de las características avanzadas que se encuentran en otros hipervisores de virtualización de metal desnudo.

Hipervisores tipo 2

- VMware Workstation / Fusion / Player

VMware Player es un hipervisor de virtualización gratuito. Está destinado a ejecutar solo una máquina virtual (VM) y no permite crear máquinas virtuales.

VMware Workstation es un hipervisor más robusto con algunas características avanzadas, como grabación, reproducción y compatibilidad con instantáneas de VM.

VMware Workstation tiene tres casos de uso principales:

1. Para ejecutar múltiples sistemas operativos.
2. Para ejecutar versiones diferentes de un sistema operativo en un escritorio.
3. Para desarrolladores que necesitan entornos de sandbox e instantáneas o para laboratorios y con fines de demostración.

- Servidor VMware

VMware Server es un hipervisor de virtualización alojado gratuito que es muy similar a VMware Workstation. VMware ha detenido el desarrollo en el servidor desde 2009.

- Microsoft Virtual PC

Esta es la última versión de Microsoft de esta tecnología de hipervisor, Windows Virtual PC y solo se ejecuta en Windows 7 y solo es compatible con los sistemas operativos Windows que se ejecutan en él.

- Oracle VM VirtualBox

La tecnología de hipervisor VirtualBox proporciona un rendimiento y características razonables si desea virtualizar con un presupuesto limitado. A pesar de ser un producto alojado gratuito con una huella muy pequeña, VirtualBox comparte muchas características con VMware vSphere y Microsoft Hyper-V.

- Red Hat Enterprise Virtualization

La máquina virtual basada en el kernel (KVM) de Red Hat tiene cualidades tanto de un hipervisor de virtualización alojado como virtual. Puede convertir el núcleo de Linux en un hipervisor para que las máquinas virtuales tengan acceso directo al hardware físico.

- KVM

Esta es una infraestructura de virtualización para el kernel de Linux. Admite la virtualización nativa en procesadores con extensiones de virtualización de hardware. El KVM de código abierto (o máquina virtual basada en el núcleo) es un hipervisor tipo 1 basado en Linux que se puede agregar a la mayoría de los sistemas operativos Linux, incluidos Ubuntu, Debian, SUSE y Red Hat Enterprise Linux, pero también Solaris y Windows .

Casos de uso de virtualización

Consolidación de servidores

El primer caso de uso de la virtualización suele ser la **consolidación de servidores**: tomar instancias separadas del servidor (una aplicación PHP, una base de datos...) y migrarlas a máquinas virtuales que se ejecutan en uno o varios servidores físicos ejecutando hipervisores.

Las empresas que implementan la consolidación de servidores experimentan una profunda transformación, pudiendo pasar de ejecutar 150 servidores físicos hasta ejecutar 150 máquinas virtuales en muchos menos servidores, lo que redunda en una **enorme reducción de costes** e inversión de hardware, energía y refrigeración, empleados, tiempo y, en muchos casos, costes de licencias de software. Por tanto, podemos afirmar que la consolidación de servidores ofrece un sustancioso porcentaje de mejora de utilización de recursos.

Entornos de desarrollo y pruebas

Vamos a imaginar que un ingeniero construye un software capaz de implementar una determinada funcionalidad. Antes de pasar a la siguiente fase, el software se ha de ejecutar y probar en una variedad de sistemas (por ejemplo, Windows y Linux), así como varias versiones de esos productos. Entonces, un grupo o departamento de calidad (QA) realiza todas las pruebas necesarias para garantizar que cumple con los requisitos aplicables de funcionalidad, escalabilidad, robustez y detectar errores.

Gracias a la implementación de la virtualización, un desarrollador o tester puede replicar un entorno distribuido con varios sistemas en una sola pieza de hardware. Esto permite librarse de la necesidad de tener un montón de servidores aparcados, esperando que los testers o desarrolladores necesiten utilizarlo.

A su vez, la virtualización también es útil en entornos de prueba y desarrollo por otro motivo. Como muchos de vosotros sabréis, uno de los efectos secundarios del software de prueba es que las primeras versiones a menudo se bloquean y dañan no solo a la aplicación, sino también al funcionamiento del sistema subyacente, así como a otras aplicaciones en la pila de software; y para poder iniciar la recuperación, es necesario reinstalar todo el software. Resulta evidente que esto representa un verdadero lastre para la productividad. Pues bien, a través de la virtualización, esto puede evitarse ya que solo la aplicación de prueba y la máquina virtual y el software relacionado con esta se ven afectados.

Computación en la nube privada

En pocas palabras, la computación en la nube es un medio para proporcionar servicios de tecnología bajo demanda a través de internet. Una nube privada es un entorno donde estos servicios operan para una única organización. Numerosas organizaciones de TI están explorando cómo construir sus propias nubes privadas para aumentar la agilidad y reducir sus costes.

La virtualización es un componente clave de las nubes privadas porque habilita el rápido **aprovisionamiento, orquestación y decomiso** de recursos y servicios bajo demanda.

Reto

Define estos tres términos, tan comunes en manejo de infraestructura, con tus propias palabras.

Calidad de servicio

Las organizaciones de TI deben centrarse en la calidad de los servicios que brindan, es decir, el buen mantenimiento de las aplicaciones y la infraestructura subyacente, así como el buen funcionamiento de estas. Cuando la virtualización se gestiona adecuadamente, puede ayudar a mejorar la calidad del servicio porque elimina la dependencia del hardware.

Además, la virtualización de los sistemas permite responder con más rapidez a todo tipo de fallos sean estos de hardware, de red o incluso si son causados por el software de virtualización. También se puede implementar de forma preventiva para evitar fallos al mover cargas de trabajo de un sistema que presenta signos de problemas inminentes (memoria, disco, etc.).

Conmutación por error

El hipervisor monitoriza constantemente el estatus de cada máquina virtual, por lo que es relativamente sencillo configurarlo para iniciar una nueva instancia de una máquina virtual en caso de que se observe que una máquina virtual que se estaba ejecutando anteriormente ya no está presente. El hipervisor simplemente tiene que iniciar una **nueva instancia**, basada en la imagen de la máquina virtual. Este proceso puede durar apenas unos segundos, y obviamente, representa una importante mejora sobre la duración típica de restauraciones de sistemas no virtualizados.

Alta disponibilidad

La alta disponibilidad (HA) extiende el concepto de conmutación por error al incorporar un servidor de hardware adicional. Es decir, que en el caso de que la máquina virtual fallase, esta no se iniciaría en la misma pieza de hardware, sino que se iniciaría en un servidor diferente, evitando así el problema de conmutación por error de virtualización por adición del hardware.

¿Cómo funciona HA?

¿Cómo es posible que un hipervisor en un servidor físico inicie una máquina virtual en otro hipervisor? La respuesta es clara: **no es posible**. No puede. La alta disponibilidad se basa en un software de virtualización global que coordina los esfuerzos de múltiples hipervisores. Cuando una máquina virtual en un servidor de hardware falla, el software de coordinación inicia una nueva máquina virtual en un servidor de hardware separado.

En realidad, es un poco más complejo que eso. El software de coordinación de virtualización monitoriza constantemente todos los hipervisores y sus máquinas virtuales. Si el software de coordinación ve que el hipervisor en un servidor ya no responde, reinicia las máquinas virtuales del hardware que ha fallado en otro hardware. Por lo tanto, la alta disponibilidad aborda el problema del fallo del hardware mediante el uso de un software de virtualización de mayor nivel para coordinar los hipervisores en dos o más máquinas, a través de la monitorización continua y el reinicio de máquinas virtuales en otras máquinas si es necesario.

La alta disponibilidad (HA, por sus siglas en inglés, High Availability) proporciona una capa adicional de protección contra la conmutación por error a través del software de virtualización. Sin embargo, la HA no proporciona la capacidad para migrar el estado actual de la memoria de la máquina virtual a la segunda máquina, de donde sacamos las siguientes conclusiones:

- HA es más útil en entornos donde la pérdida de datos de transacciones no sea posible o no acarree grandes pérdidas.
- HA no previene la pérdida de servicio al 100%.

Agrupamiento (clustering)

El agrupamiento está diseñado para garantizar que no se pierdan datos en caso de que haya un fallo de software o de hardware. El clustering ha sido ofrecido históricamente por los proveedores de aplicaciones como complemento de otros productos, pero con algunos inconvenientes relacionados: gastos adicionales, soluciones redundantes e infraestructura compleja.

El gasto extra se produce por la necesidad de contar con hardware adicional, con el sistema en espejo en espera (en modo standby), listo para asumir el control si fallase el sistema primario: un segundo conjunto de hardware hace que el agrupamiento represente un gasto significativo. Sin embargo, si en el sistema se operan transacciones de millones de euros, mantener un servidor redundante que esté listo para operar cuando haya un fallo, puede ser una inversión que valga la pena.

¿Cómo funciona el agrupamiento?

Esencialmente, el software de coordinación de virtualización ejecuta dos máquinas virtuales en máquinas separadas. Las máquinas virtuales son idénticas en cuanto al sistema operativo y la configuración de la aplicación, pero difieren, naturalmente, en los detalles de sus conexiones de red y hardware local. El supervisor de virtualización se comunica constantemente con las máquinas virtuales en el clúster para confirmar que están trabajando (**heartbeat**).

Una VM (máquina virtual) es el servidor **primario** y es el sistema con el que los usuarios interactúan. La segunda VM sirve como **backup** (copia de seguridad), lista para actuar en caso de que el servidor primario se caiga.

El desafío es que el primario envía constantemente cualquier cambio al servidor secundario para que su estado refleje el de la VM primaria en todo momento. Si la VM principal falla, el supervisor de virtualización detecta que no está disponible y cambia a los usuarios al servidor de respaldo. Los usuarios que se conecten después del cambio no notarán nada y no serán conscientes de que están conectados a una VM diferente. Por su parte, los usuarios que se han conectado a la VM original que ya no está disponible tampoco serán conscientes del cambio, porque el software de virtualización ha ido enviando el estado de estos usuarios a la máquina secundaria. A lo sumo podrán notar un descenso en la capacidad de respuesta mientras se realiza el cambio, pero, generalmente, es tan rápido que es difícil que lo detecten.

Ahora bien, como podemos ver, hay un recurso poco aprovechado en el cluster de virtualización: hay una VM que actúa como copia de seguridad y que se mantiene actualizada, pero que no realiza ningún trabajo. Aunque ejecutar una VM en un servidor virtualizado es ciertamente menos costoso que dedicar un servidor completo para actuar como una copia de seguridad activa, esto genera costes.

Duplicación de datos (data mirroring)

Hasta aquí hemos abordado los mecanismos involucrados en mantener a las máquinas virtuales en funcionamiento. ¿Pero qué hay de los datos? Después de todo, las aplicaciones dentro de las máquinas virtuales son inútiles sin datos, por lo que es claramente importante garantizar la disponibilidad de los datos como parte de una estrategia general de calidad de servicio.

Una forma de mantener los datos disponibles es a través de la duplicación. Como el nombre implica, esta duplicación o reflejo significa que los datos existentes en un sitio son reflejados en otro y ambas contienen la misma información. La duplicación permite la consistencia en tiempo real entre dos fuentes de datos. Esto posibilita el cambio inmediato entre un sistema y otro, es decir, conectando el segundo sistema a la duplicación o el reflejo de los datos del sistema primario.

Réplica de datos

La replicación es otro servicio orientado a mejorar la calidad del servicio de datos. A diferencia de la duplicación (data mirroring) que se enfoca en cómo mantener copias de datos consistentes en tiempo real, la replicación aborda la necesidad de mantener copias completas de los datos para que puedan ser utilizados en la reconstrucción del sistema. Esto se logra enviando copias de datos a un almacenamiento centralizado, lo que permite a una organización tener la seguridad de que en caso de que necesite acceder a los datos críticos por algún motivo, estos están almacenados de forma segura y disponibles en caso de ser necesarios.

La eficiencia es vital para la replicación, es decir, que no debemos pensar que porque los datos se están moviendo a una ubicación de almacenamiento hay que olvidarse de que los datos deben fluir correctamente. Un software de replicación inteligente mantiene los cambios, minuto a minuto fluyendo a la ubicación central, asegurando así que una organización de TI pueda localizar rápidamente los datos y usarlos para reconstruir el sistema en caso de fallos.

Balanceo de carga (load balancing)

El equilibrio o balanceo de carga protege a un sistema de la vulnerabilidad contra cualquier condición de error dada al implementar la denominada redundancia. Esta se logra a través de la ejecución de una o más copias de una máquina virtual en servidores separados. Cuando se ejecutan dos instancias de una máquina virtual y una de ellas se bloquea, la otra continúa funcionando. Si el hardware que da soporte a una de las máquinas virtuales falla, la otra máquina sigue funcionando. De esta manera, se evita que la aplicación sufra una interrupción.

El balanceo de carga también hace un mejor uso de los recursos de la máquina. Esto es así porque en lugar de que la segunda máquina virtual esté inactiva y no realice ningún trabajo útil, aunque esté siendo actualizada por la máquina principal, la segunda VM lleva la mitad de la carga y esto hace que al menos la mitad de sus recursos se utilicen. El uso de recursos duplicados puede extenderse más allá de las máquinas virtuales en sí mismas.

Las organizaciones que luchan por alcanzar altos niveles de disponibilidad, a menudo, implementan redes duplicadas con cada servidor físico con conexión cruzada con el resto de la red, lo que garantiza que las máquinas virtuales continuarán siendo capaces de comunicarse incluso si parte de la red se cae.

La migración hacia un almacenamiento virtualizado puede ayudar con el balance de carga: la combinación de almacenamiento y virtualización en el servidor proporciona máxima flexibilidad, mayor aprovechamiento y una administración más sencilla. Es necesario el almacenamiento en la red si las máquinas virtuales se ejecutan en servidores diferentes.

Las máquinas virtuales con carga balanceada también se pueden configurar para que funcionen como un cluster y que compartan el estado entre ellas. De esa manera, si una máquina virtual falla, su trabajo puede ser retomado por la otra máquina virtual.

Gestión de recuperación ante desastres

La recuperación ante desastres engloba a productos y procesos que ayudan a las organizaciones de TI para que puedan responder ante situaciones catastróficas, mucho peores que aquellas que desatan cuando una VM falla o aquellas en las que falla una pieza de hardware. La recuperación ante desastres entra en juego cuando todo el centro de datos se pierde, ya sea de forma temporal o permanente.

En estos casos, las organizaciones de TI necesitan luchar para mantener la infraestructura informática de toda la empresa en funcionamiento. Pensemos en el huracán Katrina: cuando se desató la catástrofe muchas empresas de TI perdieron toda la capacidad de procesamiento porque sus centros de datos quedaron completamente inundados. Como si eso fuese poco, también se perdió la conectividad a internet debido a que los centros de telecomunicaciones también estaban inundados.

En estos casos, la capacidad de reserva en el centro de datos no tiene ninguna relevancia. Ante siniestros de esta magnitud como tormentas, terremotos o desastres provocados por el hombre, se necesita contar con la capacidad de recuperación ante desastres.

En esta asignatura no podremos explorar todos los requisitos para la recuperación ante desastres, ya que es un tema demasiado extenso, pero bastará decir que es un **requisito indispensable contar con capacidad adicional en el centro de datos**, facilidad para recuperar los sistemas operativos y las aplicaciones, y una ágil administración de la infraestructura migrada. Además, se necesitará contar con un plan de contingencia para el proceso de recuperación de desastres, de modo que, si este ocurriera, el personal de TI pueda ejecutar el plan documentado y ensayado.

La virtualización será útil para la recuperación de la aplicación y las tareas de gestión: la tolerancia a fallos, la alta disponibilidad, el agrupamiento o clustering y las capacidades de virtualización de balance de carga o agrupación de servidores se pueden aplicar en un escenario de recuperación ante desastres. Todo dependerá de cuánto se desee gestionar físicamente durante el proceso de recuperación ante desastres.

Debido a que las imágenes de la VM pueden capturarse en archivos y luego iniciarse a través del hipervisor, la virtualización es una tecnología ideal para escenarios de recuperación ante desastres. Como podréis imaginar, en un momento crítico, localizar servidores físicos, configurarlos, instalar y configurar aplicaciones, hacer una copia de seguridad y actualizar el sistema puede ser una verdadera pesadilla. Además, mantener una capacidad informática adicional en un centro de datos remoto que refleje completamente la infraestructura informática primaria es extremadamente caro.

A través de la virtualización, un conjunto mucho más pequeño de máquinas puede mantenerse disponible en un centro de datos remoto, con un software de virtualización preinstalado y listo para aceptar imágenes de una VM. En el caso de que se produzca un siniestro de enormes proporciones, las imágenes de la VM se pueden transferir desde el centro de datos de producción hacia el centro de datos de *backup*. Estas imágenes de la VM pueden iniciarse con el software de virtualización preinstalado y ponerse en marcha en solo unos minutos.

En caso de dudas, si existe un riesgo inminente de que algunas transacciones se perdieran, sin que hubiese tiempo para migrar imágenes de una VM, se puede ejecutar el clúster o el balanceo de carga, a fin de que los dos centros de datos permanezcan actualizados. Esta infraestructura asegura el acceso al más valioso activo de toda organización, los datos, y, además, ayuda a las empresas a ser resilientes en caso de que se pierda el acceso a los datos en una de las ubicaciones.

Vagrant

- <https://www.vagrantup.com/intro>
- <https://devopscube.com/vagrant-tutorial-beginners/>
- <https://www.vagrantup.com/intro>
- <https://www.vagrantup.com/docs/provisioning/shell>

Cloud computing

El paradigma de la nube introduce un cambio en la visualización del sistema y los datos que son propiedad de una empresa. Además, el uso compartido de servicios o recursos, tales como el almacenamiento, hardware y aplicaciones de cloud computing, de una manera totalmente diferente ha facilitado la coherencia de los recursos y las economías de escala a través de su modelo de negocio de pago por uso.

Ya no se trata de un conjunto de dispositivos en una ubicación física que ejecutan un programa de software específico con todos los datos y los recursos presentes en un lugar físico, sino que es un sistema que se distribuye geográficamente, involucrando tanto a la aplicación como a los datos.

El desarrollo de arquitecturas y servicios distribuidos en la nube está lidiando con los mismos problemas de escalabilidad, elasticidad respecto a la demanda, acceso a la red amplia, medición de utilización, aspectos de seguridad tales como la autorización y autenticación y muchos otros conceptos relacionados con los servicios multiusuario con el fin de servir a un gran número de usuarios simultáneos en internet. La solución puede ser una nube pública, privada o híbrida, dependiendo del tipo de industria u organización del que se trate.

A través de la virtualización, un conjunto mucho más pequeño de máquinas puede mantenerse disponible en un centro de datos remoto, con un software de virtualización preinstalado y listo para aceptar imágenes de una VM. En el caso de que se produzca un siniestro de enormes proporciones, las imágenes de la VM se pueden transferir desde el centro de datos de producción hacia el centro de datos de *backup*. Estas imágenes de la VM pueden iniciarse con el software de virtualización preinstalado y ponerse en marcha en solo unos minutos.

En caso de dudas, si existe un riesgo inminente de que algunas transacciones se perdieran, sin que hubiese tiempo para migrar imágenes de una VM, se puede ejecutar el clúster o el balanceo de carga, a fin de que los dos centros de datos permanezcan actualizados. Esta infraestructura asegura el acceso al más valioso activo de toda organización, los datos, y, además, ayuda a las empresas a ser resilientes en caso de que se pierda el acceso a los datos en una de las ubicaciones.

Vagrant

- <https://www.vagrantup.com/intro>
- <https://devopscube.com/vagrant-tutorial-beginners/>
- <https://www.vagrantup.com/intro>
- <https://www.vagrantup.com/docs/provisioning/shell>

Cloud computing

El paradigma de la nube introduce un cambio en la visualización del sistema y los datos que son propiedad de una empresa. Además, el uso compartido de servicios o recursos, tales como el almacenamiento, hardware y aplicaciones de cloud computing, de una manera totalmente diferente ha facilitado la coherencia de los recursos y las economías de escala a través de su modelo de negocio de pago por uso.

Ya no se trata de un conjunto de dispositivos en una ubicación física que ejecutan un programa de software específico con todos los datos y los recursos presentes en un lugar físico, sino que es un sistema que se distribuye geográficamente, involucrando tanto a la aplicación como a los datos.

El desarrollo de arquitecturas y servicios distribuidos en la nube está lidiando con los mismos problemas de escalabilidad, elasticidad respecto a la demanda, acceso a la red amplia, medición de utilización, aspectos de seguridad tales como la autorización y autenticación y muchos otros conceptos relacionados con los servicios multiusuario con el fin de servir a un gran número de usuarios simultáneos en internet. La solución puede ser una nube pública, privada o híbrida, dependiendo del tipo de industria u organización del que se trate.

Tipos de nube

El hecho de que la información resida de forma temporal o definitiva en servidores de nube da como resultado que dichos servicios ofrezcan distintos formatos de privacidad que cada usuario puede elegir, según sus necesidades. De ahí que se planteen varios modelos de nubes como espacios de desarrollo de los servicios ofertados. Estas son:

1. Nube pública.
2. Nube privada.
3. Nube híbrida.

Nubes públicas

Los usuarios acceden a los **servicios de manera compartida** sin que exista un exhaustivo control sobre la ubicación de la información, que reside en los servidores del proveedor. Es importante resaltar que el hecho de que sean públicas no es un sinónimo de que sean inseguras, pero la realidad es que suelen ser más vulnerables a los ataques.

Cuando hablamos de nube pública, queremos decir que toda la infraestructura de computación se encuentra en las instalaciones de una empresa de cloud computing que ofrece el servicio en la nube. La ubicación permanece, por lo tanto, separada del cliente y este no tiene control físico sobre la infraestructura. Por último, como las nubes públicas utilizan recursos compartidos, se destacan principalmente por su buen rendimiento.

Nubes privadas

Nube privada significa usar **una infraestructura en la nube (red) por cada cliente u organización**. Si bien no se comparte con otros, se encuentra remotamente localizada. Las empresas tienen la opción de elegir una nube privada en la propia sede, que es más cara, pero tiene la ventaja de que así se puede tener control físico sobre la infraestructura.

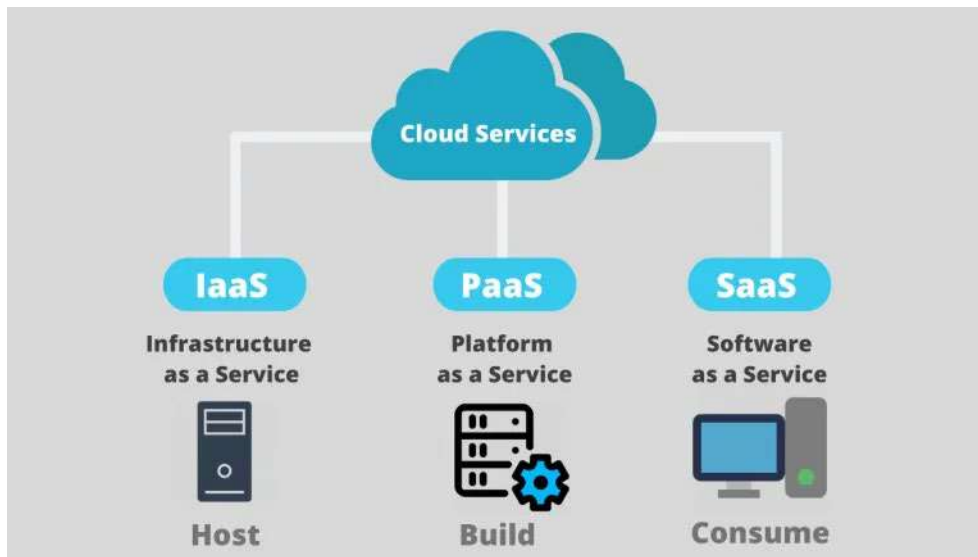
Resulta evidente que el nivel de seguridad y control es más alto cuando se utiliza una red privada que una red pública. Sin embargo, la reducción de costes puede ser mínima si la empresa necesita invertir en una infraestructura en la nube *on premise*.

Nubes híbridas

Combinan características de las dos anteriores, de manera que parte del servicio se puede ofrecer de manera privada (por ejemplo, la infraestructura) y otra parte de manera compartida (por ejemplo, las herramientas de desarrollo).

Niveles de servicio

Cuando hablamos de servicios en la nube, es importante establecer que existen diferentes opciones dependiendo del tipo de servicios que se ofrecen y la implicación que tiene el cliente en la gestión de esta. La clasificación más aceptada consiste en dividir los niveles de servicio en tres.



Overview arquitecturas cloud. Fuente: [Cloud Martial](#). (2020, marzo 1).

IaaS, PaaS and SaaS: The Definitive Guide (2020).

Infraestructura como servicio (IaaS)

También conocida como HaaS, del inglés *hardware as a service*, esta arquitectura se basa en el modelo de dotar de forma externalizada a sus usuarios / empresas del hardware necesario. IaaS proporciona hardware, almacenamiento, servidores y espacio de centro de datos o componentes de red.

Como inconveniente podemos destacar que se requiere de los mismos conocimientos informáticos en sistemas operativos y redes informáticas que necesitábamos con una arquitectura tradicional.

La idea básica es la de hacer uso externo de servidores para espacio en disco, base de datos, routers, switches... así como tiempo de cómputo evitando de esta manera tener un servidor local y toda la infraestructura necesaria para la conectividad y mantenimiento dentro de una organización. Con una IaaS lo que se tiene es una solución en la que se paga solamente por el consumo de los recursos usados: espacio en disco utilizado, tiempo de CPU, espacio para base de datos, transferencia de datos, etc.

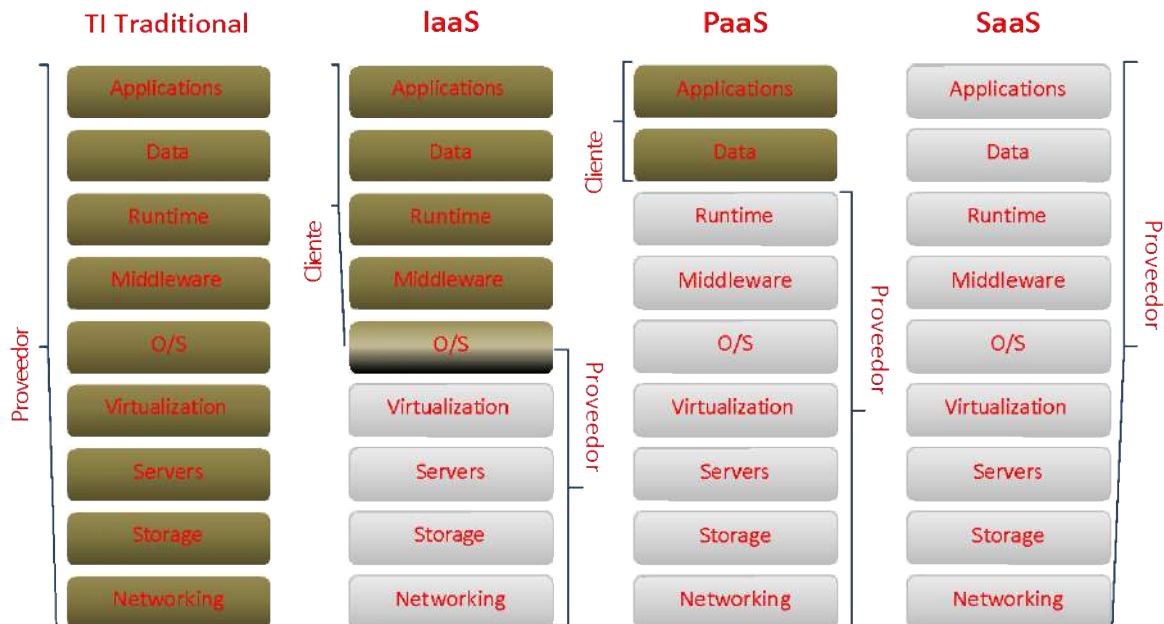
Plataforma como servicio (PaaS)

Se trata de un modelo en el que se proporciona un servicio de plataforma con todo lo necesario para dar soporte al ciclo de diseño, desarrollo y puesta en marcha de aplicaciones y servicios web a través de esta.

El proveedor es el encargado de escalar los recursos en caso de que la aplicación lo requiera, de que la plataforma tenga un rendimiento óptimo, de la seguridad de acceso, etc. Para desarrollar software se necesitan bases de datos, herramientas de desarrollo y en ocasiones servidores y redes. Con PaaS, el cliente **únicamente** se enfoca en desarrollar, depurar y probar ya que las herramientas necesarias para el desarrollo de software son ofrecidas a través de internet, lo que teóricamente permite aumentar la productividad de los equipos de desarrollo gracias a que abstrae del hardware físico al cliente.

Software como servicio (SaaS)

Consiste en la entrega de aplicaciones completas como un servicio, permitiendo la abstracción completa, no solo del hardware subyacente, sino incluso de la plataforma, permitiendo al usuario (cliente) dedicarse únicamente a su uso (consumo).



Pie: Comparativa de los niveles de servicio en la nube.

Funciones como servicio (FaaS)

Las funciones como servicio son un concepto relativamente moderno, introducido en el mundo cloud por Amazon Web Services con AWS Lambdas en el 2014.

Se trata de un tipo de servicio de computación que ejecuta código en respuesta a eventos, pero sin que los usuarios se tengan que preocupar de la infraestructura que sería necesaria para albergar un servicio de estas características: esto permite a los desarrolladores enfocarse únicamente en la creación de código fuente funcional.

FaaS es un concepto que se suele asociar, o incluso equiparar, con otro paradigma de computación cloud: **serverless**.

El concepto *serverless* es el de un PaaS para aplicaciones orientadas a internet o web, en el que el coste es por llamada a la aplicación. Esto significa que, si no tiene llamadas, el servicio se apaga y queda en espera automáticamente. Y, en caso de tenerlas, el servicio escala automáticamente y de manera transparente para atender la demanda.

Pero además, *serverless* sí que incluye toda la tecnología subyacente para dar servicio a las FaaS, por lo que deben considerarse estas como un subtipo dentro del mundo *serverless*.

Proveedores de nube pública

En la actualidad, existen múltiples proveedores de nube pública que intentan repartirse el mercado de los servicios en la nube, sin embargo, existe un claro líder tanto de mercado como por número de servicios ofrecidos, regiones donde se pueden contratar y capacidad total, ese líder es Amazon Web Services, también conocido como AWS.

Existen otros proveedores también con amplias capacidades y que siguen de cerca al líder, en este caso hablamos de Google con su nube GC y Microsoft con Azure.



Cuadrante mágico. Fuente: Magic Quadrant 2019. Gartner, Inc.

¿Qué es AWS?

Amazon Web Services, en adelante AWS, es una plataforma de servicios en la nube que provee un gran número de servicios de infraestructuras, tales como almacenamiento, comunicaciones, bases de datos, macrodatos (big data), aprendizaje automático (machine learning), servicios de movilidad, soluciones empresariales prepaquetizadas y certificadas, securización y gestión de identidades, entre otras. Dichas infraestructuras proporcionan a las empresas un crecimiento y control de gastos a medida y, sobre todo, aportan escalabilidad.

La nube de AWS incluye 81 zonas de disponibilidad en 25 regiones geográficas de todo el mundo, a noviembre de 2021. Además, tienen planes para incorporar 24 zonas de disponibilidad y 8 regiones de AWS adicionales en Australia, India, Indonesia, Israel, Nueva Zelanda, España, Suiza y Emiratos Árabes Unidos (EAU).

En casi todas las regiones AWS ofrece los mismos servicios. El número de servicios disponible es **inmenso** y abarca todas las aplicaciones que se nos puedan ocurrir. Por eso, vamos a destacar los más comunes ya que serán los que, de manera más recurrente, nos encontremos en la práctica.

Para información sobre todo lo que nos podemos encontrar, consultad la [hoja de productos de AWS](#).

Clouds privadas virtuales (VPC)

Una VPC es necesaria para poder lanzar la gran mayoría de los recursos de AWS que necesiten una infraestructura de red por debajo como, por ejemplo, las máquinas virtuales de EC2, bases de datos...

Es un servicio **complejo**, cuya configuración adecuada exige conocimientos de redes que aún no tenemos. A pesar de eso, existen asistentes y configuraciones que podemos adoptar (por ejemplo, [un caso sencillo](#)) gracias a las que podremos empezar a trabajar sin problemas.

Conceptos relacionados con VPC que debemos conocer:

- **Subred:** un intervalo de direcciones IP en la VPC definido por nosotros según las necesidades que hayamos detectado durante la planificación.
- **Tabla de enrutamiento:** un conjunto de reglas, denominadas rutas, que se utilizan para determinar dónde se dirige el tráfico de red.
- **Gateway de internet:** una gateway que asocia a la VPC para habilitar la comunicación entre los recursos de la VPC e internet (sin acceso a internet, no tendremos manera de entrar en máquinas que hayamos levantado sin usar la consola).
- **Punto de enlace de la VPC:** permite conectar de manera privada la VPC a los servicios admitidos de AWS y a los servicios del punto de enlace de la VPC habilitados por PrivateLink, sin necesidad de contar con una puerta de enlace de internet, un dispositivo NAT, una conexión de VPN ni una conexión de AWS Direct Connect. Para obtener más información, consultad [AWS PrivateLink y los puntos de enlace de la VPC](#).
- **IP elástica:** las direcciones IP elásticas son direcciones IPv4 estáticas y públicas, asociables con cualquier instancia o interfaz de red de cualquier VPC de la cuenta, incluso 'en caliente' para reapuntar el tráfico público cuando se detecte un error.

Elastic cloud computing (EC2)

Es el servicio de virtualización de hardware de AWS, en el que lanzaremos diferentes tamaños de máquinas (instancias) optimizadas para memoria, rendimiento u operaciones de E/S, con los posibles sistemas operativos que provee Amazon (algunos de pago) o imágenes que creamos nosotros y subamos a nuestra cuenta (AMI: Amazon machine image).

Estas instancias pueden crearse y destruirse con gran velocidad, desde multitud de servicios de AWS de forma indirecta o directamente desde la consola o la línea de comandos. Conceptos que podremos configurar o asociar a las instancias y que debemos conocer son:

- **Elastic Block Storage:** el almacenamiento de las instancias es elástico, en cuanto a que debemos especificar en tiempo de creación qué tipo y tamaño tendrá, aunque se pueden añadir más discos en caliente o incluso redimensionar los que ya estén montados.
- **Security Groups:** los grupos de seguridad se pueden asemejar a un set de reglas de un Firewall, ya que determinan desde qué IPs, o rangos de IPs, y a través de qué puertos se permite el acceso a nuestras instancias. En muchas ocasiones, AWS nos ayudará a configurar el acceso entre servicios, pero necesitarán configuración manual en la mayoría de los casos.
- **Claves SSH:** nos las pedirá AWS en la creación de cada instancia, ya que sin ellas es imposible conectarse a una máquina ya levantada para su configuración o mantenimiento, salvo que tenga configurada alguna UI en arranque a la que podamos llegar con un Security Group.

Simple storage service (S3)

Es un servicio de almacenamiento de datos de muy alto rendimiento y capacidad prácticamente ilimitada. Existen APIs de acceso desde casi cualquier lenguaje que permiten usarlo como un sistema de ficheros más, con la ventaja que ofrece la ubicuidad de acceso.

Es muy común que otros servicios de AWS utilicen los buckets de S3 para almacenar datos intermedios como, por ejemplo, AWS CloudFormation. O que exista alguna integración nativa desde un PaaS, como el servicio de AWS Lambda.

Como curiosidad: AWS ha publicado un SLA para S3 de 11 (¡once!) nueves, lo que se traduce en que se compromete a que el servicio S3 esté disponible un 99.999999999% del tiempo.

Reto

¿Cuántos segundos al año se permite de *downtime* en S3? ¿Y si fuese un servicio de 5 nueves? ¿y de 3?

Lambda

Lambda es un FaaS, una variante del PaaS que ya vimos en la sección de cloud. Nos permite ejecutar código en respuesta a eventos sin tener que preocuparnos por gestionar ninguna infraestructura por debajo: ni instancias, ni permisos, ni redes...

Es por esto que el coste es mucho menor que si tenemos que hacerlo por nuestra cuenta y, además, con la ventaja de que AWS se encarga de escalar el servicio de forma transparente para atender a toda la demanda, en caso de picos de tráfico.

Relational database service (RDS)

Con Amazon Relational Database Service (Amazon RDS) es sencillo configurar, utilizar y escalar una base de datos relacional en la nube. El servicio suministra capacidad rentable y escalable al mismo tiempo que automatiza las arduas tareas administrativas, como el aprovisionamiento de hardware, la configuración de bases de datos, la implementación de parches y la creación de copias de seguridad.

Amazon RDS se levanta sobre EC2 y, por tanto, tendremos la posibilidad de optimizar para memoria, rendimiento u operaciones de E/S. El servicio nos proporciona seis motores de bases de datos entre los que elegir: [Amazon Aurora](#), [PostgreSQL](#), [MySQL](#), [MariaDB](#), [Oracle Database](#) y [SQL Server](#).

CloudWatch

Es un servicio que agrega en un panel datos otros servicios de AWS como métricas, logs y alarmas. Sobre estos datos, se pueden generar paneles de control, o se pueden tomar acciones en función de esas alarmas, lo que supone la base del autoescalado de EC2, que permite crear o destruir instancias en base a métricas complejas.

Todo esto lo convierte en una funcionalidad **esencial** que debemos conocer y manejar con soltura, pues la alternativa sería integrarnos con un stack como Logstash o Fluentd, con su back-end propio y su propio servicio de visualización (Kibana).

Esto no sería nada malo, pues son herramientas con muchísima potencia, con lo que volveremos a considerar si nos cuesta menos montar y mantener una solución ad hoc o consumir el servicio gestionado por AWS.

Identity and access management (IAM)

Este servicio es, sin duda, el más importante del stack de servicios de AWS, ya que se encarga de gobernar el control de acceso a nuestra cuenta y a todos los servicios desplegados en ella: mediante el uso de políticas de acceso (*AWS Policies*) se permite especificar con detalle **qué puede hacer cada usuario**, desde desplegar cualquier instancia en cualquier región a tener tan solo acceso a logs en una región determinada.

El manejo avanzado de IAM es algo que cualquier administrador IT debe conocer, pues de un uso acertado de los permisos depende la seguridad de la cuenta y de todos los usuarios que la manejan. Esto implica muchas consideraciones a tener en cuenta desde el punto de vista de la seguridad, algunas de las más importantes son:

- Asignar a cada usuario los permisos **mínimos necesarios**, incluso a costa de ir iterando de menos a más hasta que pueda hacer su trabajo. Hacerlo en el sentido opuesto (ir quitando políticas de acceso más amplias) pone en riesgo la seguridad de la cuenta.
- Organizar a los usuarios en **grupos**, donde se pueden aplicar políticas de forma transversal y evitar el tener que manejar casos uno a uno. Los tipos de grupos usuales son developers, testers, gestores de equipo...
- **Rotar** las claves y contraseñas con frecuencia, tanto de acceso a la consola como programático o usar MFA (*Multi-factor authentication*) para aumentar la seguridad.

Todas estas consideraciones se pueden automatizar o, al menos, facilitar utilizando AWS IAM. Además, existen multitud de vídeos y tutoriales de Amazon que permiten aprender a manejar con soltura una herramienta que, sin duda, merece la pena aprender si se va a utilizar AWS: [Introducción a AWS Identity and Access Management \(IAM\)](#).

Acceso gratuito a AWS

Existe la posibilidad de crear una cuenta de AWS gratuita, que sirve perfectamente para los propósitos del curso. Los detalles se pueden ver aquí: <https://aws.amazon.com/es/free/>.


El nivel gratuito de AWS ofrece a los clientes la posibilidad de explorar y probar los servicios de AWS sin costo hasta llegar a los límites especificados para cada servicio. La capa gratuita consta de tres tipos diferentes de ofertas, una capa gratuita de 12 meses, una oferta siempre gratis y las pruebas a corto plazo.

- Los servicios con una capa gratuita de 12 meses permiten a los clientes utilizar el producto gratis durante un año (a partir de la fecha en que se creó la cuenta) hasta alcanzar los límites específicos.
- Los servicios con una oferta siempre gratis permiten a los clientes utilizar el producto de forma gratuita hasta alcanzar los límites especificados, siempre que sean clientes de AWS.
- Los servicios con una versión de prueba a corto plazo se pueden usar gratis durante un período de tiempo específico o se pueden usar una sola vez, según el servicio seleccionado.

¡Importante!

Hay que tener en cuenta que para darse de alta hay que añadir datos de facturación, ya que en el caso de que se usen recursos que no entran en la capa gratuita, Amazon cobrará el gasto incurrido.

Verificación segura

-  No cobraremos el uso que esté por debajo de los límites de la capa gratuita de AWS. Retenemos temporalmente 1 USD/EUR como transacción pendiente por un periodo de 3 a 5 días para verificar su identidad.



Registrarse en AWS

Información de facturación

Número de tarjeta de crédito o débito



AWS acepta todas las tarjetas de crédito y débito principales. Para obtener más información sobre las opciones de pago, consulte nuestras [preguntas frecuentes](#)

Fecha de vencimiento

Nombre del titular de la tarjeta

Dirección de facturación

Fuente: <https://www.amazon.es/>

El nivel gratuito de AWS está disponible para todas las cuentas de clientes: estudiantes, emprendedores, pequeñas empresas y empresas Fortune 500. Todos pueden registrarse.

Esta oferta proporciona a los nuevos clientes de AWS niveles de uso gratuitos en determinados servicios de AWS para ayudarles a ponerse en marcha. Es una forma fantástica de comenzar de forma gratuita a probar los servicios que ofrece AWS o poner en marcha el alojamiento de sitios web o blogs con poco tráfico, aplicaciones para redes sociales, proyectos de desarrollo y prueba, pruebas de concepto, etc.

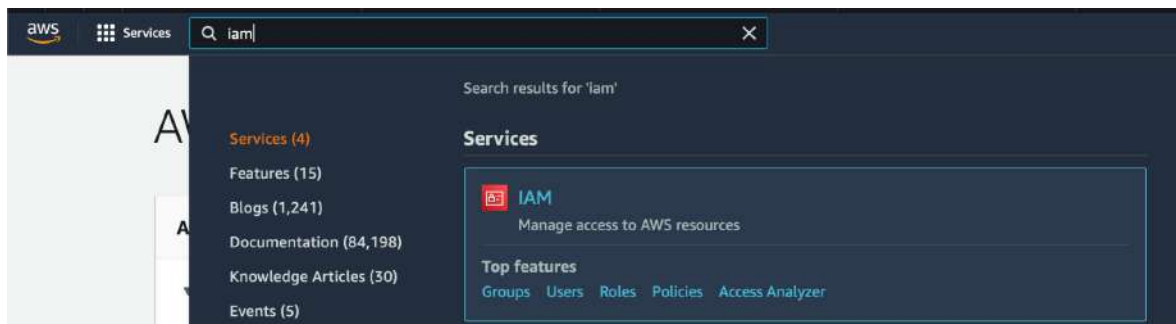
Acceso programático a AWS

A pesar de que la consola de AWS es una aplicación web potentísima desde la que es posible realizar **absolutamente todas** las operaciones que necesitaremos para prácticamente cualquier aplicación, como responsables de operaciones siempre preferimos poder utilizar herramientas que, de forma programática, nos den acceso a las mismas funcionalidades.

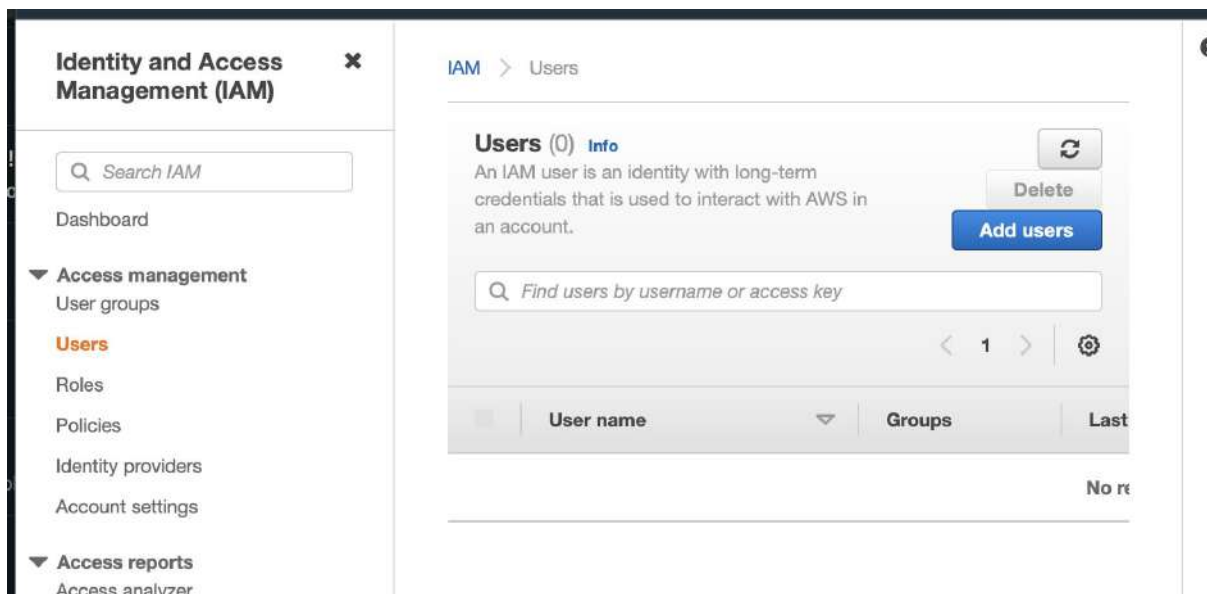
Como veremos, para poder usar cualquier herramienta que interactúe con AWS desde un terminal, necesitaremos tener unas claves de acceso que nos lo permitan.

Los pasos para conseguir esto son:

1. En la [consola de AWS](#), buscar el servicio de gestión de identidades y acceso (IAM):



2. Crear un usuario IAM nuevo. En la selección de tipo de credencial AWS, seleccionar tipo '**Programmatic Access**':



- Si no existen grupos, cread uno nuevo, con permisos de administrador. (**Esto no es una buena práctica de cara a futuro**, los permisos deben ser los mínimos imprescindibles).

Add user

1 2 3 4 5

Set permissions

 Add user to group

 Copy permissions from existing user

 Attach existing policies directly

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

Add user to group

Create group Refresh

Search		Showing 1 result
Group	Attached policies	
<input checked="" type="checkbox"/> admin-group	AdministratorAccess	

Set permissions boundary

Set a permissions boundary to control the maximum permissions this user can have. This is an advanced feature used to delegate permission management to others. [Learn more](#)

- ☒ Create user without a permissions boundary
☐ Use a permissions boundary to control the maximum user permissions

- Finalizar el proceso. No necesitamos tags por ahora: **guardad el fichero y apuntad la clave secreta**, después de esto no podremos recuperarla sin ese fichero.

Una vez tengamos el 'AWS_ACCESS_KEY_ID' y el 'AWS_SECRET_ACCESS_KEY', debemos guardarlos en alguna parte de nuestro sistema. Las ubicaciones usuales son:

- ~/aws/credentials: Es la ubicación recomendada si se van a usar las herramientas de AWS de línea de comandos ([aws-cli](#)) o las APIs para AWS de diferentes lenguajes (como [boto3](#) para Python).
- ~/profile (o similares): Es la ubicación usual para variables de entorno en un sistema Unix/Linux, aunque hay que tener cuidado si es un sistema compartido para no exponerlas a otros usuarios.
- Como variables de entorno, invocadas explícitamente en cada llamada (incómodo e inseguro, pero útil en scripts y Dockerfiles).

Llamada errónea

```
$ AWS_ACCESS_KEY_ID=xxx AWS_SECRET_ACCESS_KEY=yyy aws s3 ls
```

An error occurred (InvalidAccessKeyId) when calling the ListBuckets operation: The AWS Access Key Id you provided does not exist in our records.

Llamada con éxito (no devuelve nada porque no hay nada en los buckets)

```
AWS_ACCESS_KEY_ID=tu_key AWS_SECRET_ACCESS_KEY=tu_secret aws s3 ls
```

Infraestructura como código

Hasta hace unos años, los responsables de sistemas tenían que montar sus infraestructuras de hardware y software de forma manual. Es decir, que montaban y apilaban el hardware para, posteriormente, instalar y configurar los sistemas operativos y aplicaciones necesarias para el negocio. Como podréis imaginar, este proceso, además de ser muy tedioso, normalmente se ejecutaba por más de una persona, lo que generaba una larga espera hasta que la infraestructura de aplicaciones y sistemas estuviera disponible.

Para agilizar estas implementaciones, optimizar el rendimiento y entregar con más rapidez el software desplegado en las plataformas, los responsables de sistemas crearon scripts para unificar, automatizar y optimizar el proceso.

A su vez, la aparición de las metodologías ágiles ha instaurado una nueva forma de gestionar el software, que implica que la instalación, configuración y mantenimiento de los sistemas informáticos se haga más compleja, si cabe. Estas metodologías requieren de varios ambientes donde ejecutar las aplicaciones (desarrollo, testing, staging, etc.), lo cual termina duplicando, al menos, el esfuerzo de los administradores.

La respuesta que ha dado la informática a estos problemas es la optimización y automatización del despliegue de aplicaciones y la configuración de los servidores. Actualmente, es posible configurar servidores a través de la programación y sin intervención de los administradores. Esta nueva forma de administración, que considera a la infraestructura como un aplicativo más a ser gestionado de manera análoga al software, se la conoce como infraestructura programable o **infraestructura como código**, más conocida como **IaC** (Huttermann, M., 2012).

La infraestructura como código trata la infraestructura de configuración de sistemas como un software de programación. Esto genera una delgada línea entre los límites de la escritura de aplicaciones y la creación de entornos en los que se ejecutan. Se trata de una **parte fundamental** de la computación en la nube y esencial para DevOps. La IaC es el marco que ha dado origen a DevOps.

Diferencias con a la operación manual

El trabajo tradicional del operador de sistemas tiene una serie de **problemas sistémicos**, comúnmente asociados al factor humano:

- **Alto coste**, ya que necesitan un significativo y cualificado capital humano por parte de la organización.
- **Inconsistencias**, debido a errores humanos y a la dificultad de evitar, derivan de las configuraciones en sistemas a lo largo del tiempo.
- **Falta de agilidad**, al limitar la velocidad a la que las organizaciones pueden lanzar nuevas versiones en respuesta al mercado.
- **Dificultad** en alcanzar y mantener estándares de la industria, debido a la falta de o a la complejidad de procesos de configuración.

La IaC ofrece las siguientes **ventajas** frente a la configuración manual tradicional:

- **Alta eficiencia:** Automatiza la mayor parte de la administración de los recursos, lo que lleva a optimizar el ciclo de vida de desarrollo SW.
- **Reutilización:** Una vez se haya descrito una infraestructura como código, esta se puede ejecutar en cualquier momento, todas las veces que se desee, de forma **idempotente**.
- **Control de versiones:** Al ser código fuente, lo natural es que se almacene en un repositorio, lo que lleva a poder revisar los cambios a lo largo del tiempo.
- **Minimiza costes y esfuerzo:** Al automatizar trabajo tedioso y tendiente a errores.

Ciclo de vida de los recursos de infraestructura

Las etapas del ciclo de vida de los recursos son las siguientes:

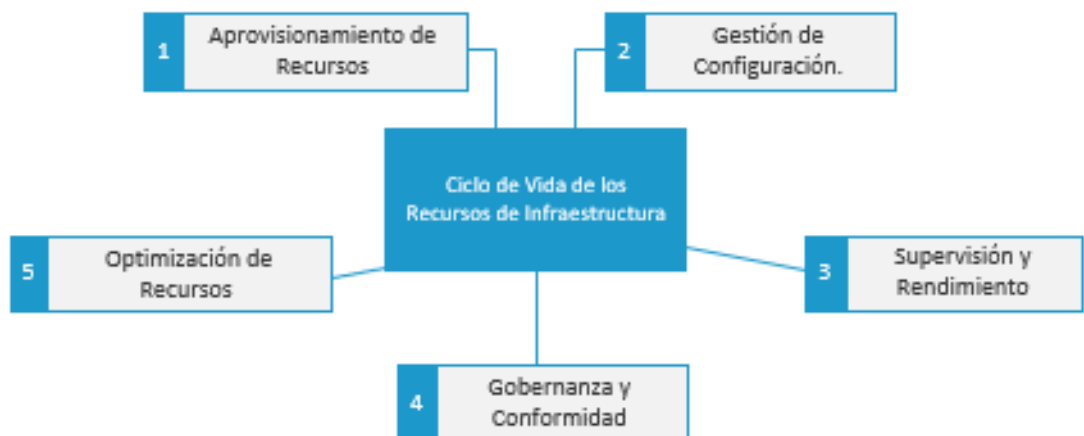


Figura 2. Ciclo de vida de los Recursos de Infraestructura

fuelle: <https://docs.google.com/document/d/1Li9gkNfBd9RnHjLUIFrLjKylReR6itE/edit#>

Cada etapa involucra procedimientos que pueden aprovecharse en código, lo cual extiende los beneficios de la IaC de su rol tradicional de aprovisionar al ciclo de vida entero. Cada ciclo de vida se beneficia de la consistencia y repetibilidad que la infraestructura como código ofrece.

1. **Aprovisionamiento de recursos.** Etapa en la que los administradores utilizan tecnologías de IaC para describir y aprovisionar recursos cloud de acuerdo con sus especificaciones y necesidades. AWS CloudFormation o Terraform son ejemplos de tecnologías en este punto.
2. **Gestión de configuración.** Los recursos se vuelven componentes de un sistema de gestión de configuración que soportan actividades tales como optimización y actualización.
3. **Supervisión y rendimiento.** Herramientas de supervisión y rendimiento validan el estado operacional de los recursos examinando ítems como métricas, transacciones sintéticas y archivos de registro.

4. **Gobierno y conformidad.** Los marcos de trabajo de conformidad y gobierno gestionan la validación adicional a fin de asegurar la relación y consonancia con la corporación y los estándares de la industria, así como requerimientos regulatorios.
5. **Optimización de recursos.** Los administradores revisan datos de rendimiento e identifican cambios necesarios para optimizar el ambiente alrededor de los criterios tales como rendimiento y los costes.

Herramientas

Existen bastantes tecnologías en el mercado que se encargan, de diferentes formas, de solucionar el problema de cómo describir recursos cloud, con sus diferentes parámetros y las relaciones entre ellos.

Independientemente de la herramienta que usemos, será necesario trabajar con algún **lenguaje para la descripción de la configuración**, pudiendo ser Json, Yaml o algún nuevo lenguaje propio del software. Dentro de los archivos de configuración es donde definiremos todos los componentes necesarios para crear nuestra infraestructura.

Existen dos grandes grupos de herramientas de IaC:

Herramientas de proveedor

Son las herramientas presentes en los grandes proveedores de la nube, y permiten describir y provisionar únicamente sus propias tecnologías.

Los tres ejemplos más conocidos son:

- AWS CloudFormation

CloudFormation es la herramienta interna de IaC de Amazon Web Services (AWS) y, como tal, es prácticamente imprescindible para cualquiera que trabaje con productos de AWS como ELB, S3 o EFS. Utilizarla no conlleva ningún coste adicional, tan solo hay que pagar por los recursos reservados.

- Azure Resource Manager

Servicio de Azure, plataforma en la nube de Microsoft, que proporciona la administración de infraestructuras mediante plantillas, de forma que implementa y supervisa todos los recursos. Esto da coherencia a la hora de reimplementar recursos ya existentes y permite definir las dependencias de estos.

- Google Cloud Deployment Manager

Deployment Manager es para la plataforma Google Cloud lo que CloudFormation es para AWS. Con esta herramienta gratuita, los usuarios de recursos de IaaS de Google pueden administrarlos fácilmente mediante archivos de configuración central en el lenguaje de marcado YAML.

Herramientas multiprovider

Permiten administrar recursos de múltiples proveedores y eliminan la necesidad de conocer las API correspondientes en profundidad. Como contrapartida, pueden ser más complicados de manejar y la interoperabilidad total entre proveedores no está plenamente garantizada.

Las tecnologías multiproveedor más empleadas actualmente son:

- Terraform

Terraform es una herramienta que se utiliza para la construcción, el cambio y el versionado de infraestructura, de manera segura y eficiente. Esta puede administrar tanto servicios existentes como nubes públicas o soluciones internas personalizadas.

- Heat

Implementa un motor de orquestación para poder lanzar múltiples aplicaciones en la nube basadas en plantillas, proporcionando una compatibilidad total con las plantillas de AWS CloudFormation. Proporciona, a su vez, una API nativa y una API compatible con AWS CloudFormation. Heat es el proyecto más ambicioso de [OpenStack](#).

- Chef Infra

Chef Infra, la solución de IaC de la empresa estadounidense Chef, está disponible desde abril de 2019 bajo la licencia gratuita Apache 2.0 y es utilizado por Facebook, entre otras empresas. Entre las plataformas compatibles se incluyen Google Cloud, Microsoft Azure, Amazon EC2 y OpenStack.

- Puppet

Herramienta Open Source de gestión desarrollada en Ruby para la administración de sistemas de forma declarativa, la cual al estar basada en modelos no requiere un alto conocimiento de programación para su uso.

- Red Hat Ansible Tower

La herramienta de infraestructura como código de Ansible forma parte del catálogo de desarrollo de software Red Hat desde el año 2015. Ofrece un panel de control, su propia línea de comandos y una potentísima API REST. En este caso, ambos paquetes disponibles, tanto el estándar como el extendido, son de pago.

Mapa de herramientas IaC



Pie: Mapa de herramientas IaC. Fuente: [EdynTechnology](#). (s. f.). Infrastructure as Code (IaC).

Terraform

Terraform es una herramienta de **código abierto** creada por HashiCorp y escrita en el lenguaje de programación Go. El código Go compila en un solo binario (o más bien, un binario para cada uno de los sistemas operativos admitidos) llamado Terraform.

Este binario nos permitirá implementar infraestructura desde un ordenador o servidor de compilación con mínima intervención manual y sin necesidad de incluir una infraestructura adicional. Eso es porque el binario Terraform realiza llamadas API en su nombre a uno o más proveedores, como Amazon Web Services (AWS), Azure, Google Cloud, Digital Ocean, OpenStack, etc. Eso significa que Terraform aprovecha los proveedores de infraestructura que ya tiene para interactuar con infraestructura mediante APIs, por ejemplo, los mecanismos de autenticación que ya tiene por estar conectado a esos proveedores (con AWS utiliza las claves que ya tiene).

El código de Terraform tiene una sintaxis simple, característica que resulta muy agradecida para los usuarios. Además, permite implementar recursos interconectados en múltiples proveedores de la nube, así que podremos implementar infraestructura completa — servidores, bases de datos, equilibradores de carga (load balancers), topología de red, etc.— en los archivos de configuración de Terraform y **asignar esos archivos al control de versiones**.

Sus comandos, como, por ejemplo, **'terraform apply'**, nos permitirán implementar esa infraestructura. La tarea que lleva a cabo el binario de Terraform es la de analizar el código, traducirlo en llamadas API a los proveedores de nube que se hayan especificado en el código y, de la forma más eficientemente posible, realizar dichas llamadas a la API.

Cuando necesites realizar cambios en la infraestructura, en lugar de actualizar la infraestructura de forma manual en los servidores, puedes directamente realizar los cambios en los archivos de configuración de Terraform. Esos cambios se validarán mediante pruebas automáticas y revisiones de códigos, se confirmará el código actualizado control de versiones y luego se ejecutará el comando **'terraform apply'** para que Terraform realice las llamadas API necesarias para implementar los cambios.

Los archivos de configuración indican a Terraform cuáles son los componentes necesarios para el despliegue desde una única aplicación o bien, desde el centro de datos completo. Terraform genera un plan de ejecución con una descripción completa sobre aquello que hará para alcanzar el estado deseado y, a continuación, lo ejecuta para que la infraestructura descrita pueda ser construida. Terraform tiene la capacidad de determinar qué ha cambiado, a medida que las modificaciones tienen lugar, y además, elaborar planes de ejecución incrementales para que, posteriormente, se ejecuten.

La infraestructura que Terraform puede administrar incluye componentes de bajo nivel como instancias de cómputo, almacenamiento y redes, así como componentes de alto nivel tales como entradas de DNS, funciones de SaaS, etc. Como sabemos, es posible gestionar el propio código de una aplicación, pero el ciclo de vida del código de una aplicación suele ser más complejo y, por ello, es conveniente usar otras herramientas además de Terraform.

Características clave de Terraform

Veremos a continuación las características fundamentales que hacen de Terraform una herramienta altamente eficaz:

- Infraestructura como código

La infraestructura se describe utilizando una sintaxis de configuración de alto nivel. Esto permite que un plano (blueprint) del centro de datos pueda ser versionado y tratado como cualquier otro código. Además, la infraestructura se puede compartir y reutilizar.

- Planificación de la ejecución

En el paso de 'planificación' Terraform genera un plan de ejecución. Este plan muestra qué hará Terraform cuando ejecute la llamada a apply. De esta manera, es posible anticipar cualquier movimiento inesperado mientras Terraform manipula la infraestructura.

- Grafo de recursos

Terraform crea un grafo de todos los recursos y paraleliza la creación y modificación de cualquier recurso no dependiente. Por eso, podemos decir que Terraform construye la infraestructura de la manera más eficiente posible, a la vez que ofrece una visión de las dependencias en su infraestructura.

- Automatización de cambios

Es posible añadir conjuntos de cambios complejos a la infraestructura sin apenas realizar trabajo manual. Esto es debido al plan de ejecución y el grafo de recursos que hemos visto antes, que aportan información exacta sobre qué cambiará Terraform y en qué orden lo hará. Como es de esperar, esto evita errores humanos y agiliza el trabajo de forma considerable.

AWS CloudFormation

Vamos a ver la diferencia en el concepto de IaC cuando usamos una solución de un proveedor cloud. Concretamente, vamos a ver qué nos ofrece AWS CloudFormation, cuáles son las diferencias y similitudes con Terraform, y un pequeño ejemplo de la sintaxis que nos podemos esperar encontrar en las plantillas.

- Formatos válidos.
- Templates.
- Stacks.
- Changesets.

Fuente: [Introducción a AWS CloudFormation - Amazon Web Services](#)

Nota

Si no tienes un par de claves de Amazon EC2, debes crear el par de claves en la misma región donde se está creando la pila. Para obtener información sobre cómo crear un par de claves, consulta [cómo obtener un par de claves SSH](#) en la guía del usuario de Amazon EC2 para instancias de Linux.

Referencias bibliográficas

- Alvarado, G. (2019). Tutorial: *Infraestructura como código con Terraform*. Recuperado de <https://galvarado.com.mx/post/tutorial-infraestructura-como-c%C3%B3digo-con-terraform/>.
- Castillo Cotán, J. M. (2017). *Casos de uso — documentación de Terraform: Infraestructura como código - 3.0*. Recuperado de <https://terraform-infraestructura.readthedocs.io/es/latest/casosdeuso/>.
- Documentación oficial de Terraform: <https://www.terraform.io/docs/index.html>.

unir LA UNIVERSIDAD
EN INTERNET | FORMACIÓN
PROFESIONAL

PROEDUCA