MÁQUINAS VIRTUALES

ESCRITORIOS REMOTOS

- Son programas que nos permiten acceder e interactuar con una computadora a distancia a través de una conexión a internet permitiendo que podamos trabajar desde cualquier lugar.
- Estos programas no requieren de una red de conexión física ni un hardware adicional para vincular los computadores. Solo se necesita:
    - Acceso a internet
    - Que ambas computadoras tengan la misma aplicación de escritorio remoto
    - Que ambas computadoras permanezcan encendidas de manera simultánea
- La computadora a la que se accede de forma remota recibe el nombre de **host**, mientras que la computadora desde la que se va a trabajar físicamente se conoce como **cliente**
    - Es posible que varios clientes accedan a un mismo host, mientras este cuente con la capacidad suficiente para soportar todas las conexiones simultáneas
- Una de sus **ventajas** es el ahorro de recursos
    - Mejor calidad de vida (no tenemos que transportarnos hasta la oficina)
    - Mayor productividad
- La principal **desventaja** es que si el programa que brinda el servicio de escritorio remoto no posee la seguridad necesaria puede ser objeto de ciberataques, cuyo objetivo es robas las credenciales de las cuentas para utilizar el host
    - Otra **desventaja** es que el rendimiento depente enteramente de la calidad de conexión a internet. Si esta no es confiable, puede comprometer a todo el sistema
- Existen varias aplicaciones que ofrecen este servicio. Una de las más usadas es **Team Viewer**, por su facilidad de uso, compatibilidad con múltiples plataformas y una opción gratuita para uso personal
    - Otras son:
        - AnyDesk
        - O Assist
        - Chrome Remote Desktop
        - Windows Remote Desktop

MÁQUINA VIRTUAL

Es un software capaz de contener en su interior un sistema operativo haciéndole creer que es una computadora de verdad. Ese sistema operativo puede albergar, a su vez, otro más.

Existen dos tipos:

- Máquina virtual de sistemas
    - Emula a una computadora completa
    - Es un software que nos permite ejecutar otro sistema operativo en su interior
    - El lugar donde la máquina es creada se llama **Hipervisor**, el cual es una capa de software que se instala sobre la parte física de la computadora (hardware) y su función es asignar parte de la memoria (RAM), disco duro, CPU y otros recursos físicos
        - Existen dos tipos de hipervisor:
            - Tipo 1
                - Es el más utilizado por ser más rápido y seguro
                - Corre directamente sobre la parte física de la computadora y sobre él se crearán una o más máquinas virtuales
                - Ejemplos:
                    - VMware ESXi
                    - Microsoft Hyper-V
            - Tipo 2
                - Corre sobre un sistema operativo y es más lento
                - Ejemplos:
                    - Oracle Solaris Zones
                    - Oracle VM Server para x86
                    - Oracle VM Virtual Box
                    - VMware Workstation
                    - VMware Fusion
        - Sobre el hipervisor podemos crear tantas máquinas virtuales como queramos y cada una funciona como una computadora real
- Máquina virtual de procesos
    - No emula una computadora completa, sino solo un proceso concreto (por ejemplo, una aplicación), permitiendo que cada uno se comporte de la misma manera independientemente del sistema operativo sobre el que se ejecute, lo cual es de utilidad al momento de desarrollar aplicaciones que van a ejecutarse en distintos sistemas operativos

Ventajas:

- Podemos probar otros sistemas operativos sin cambiar el hardware
- Es posible ejecutar programas antiguos
- Permite ejecutar aplicaciones de otros sistemas operativos
- Nos ofrece un entorno de seguridad para analizar cómo funcionan virus y malwares

- Especialmente en servidores, su uso permite mejorar el aprovechamiento físico (hardware) al utilizar los recursos que de otra forma estarían ociosos, porque en general nunca se llegan a utilizan los recursos de un servidor físico al mismo tiempo
- Permite llevar la virtualización a otra áreas, como son el almacenamiento o las redes

Desventajas:

- Son menos eficientes que las máquinas reales porque acceden al hardware de forma indirecta, ya que el software se ejecuta sobre el sistema operativo de la máquina virtual y tiene que solicitar acceso al hardware de la máquina física, ralentizando el proceso
- Cuando varias máquinas virtuales ejecutan la misma máquina física el rendimiento puede verse afectado si la computadora carece de los recursos suficientes

CONTENEDORES

Un **contenedor** es un concepto de empaquetación (espacio virtual) de software que incluye la aplicación y todas sus dependencias de ejecución (herramientas) Contiene:

- Software
- Librerías
- Sistema operativo

Su objetivo es que el cliente pueda abrir el contenedor en cualquier sistema operativo y pueda ejecutar el software sin problemas

- Trabajan de manera conjunta con el sistema operativo ya que no requieren de un hipervisor, por lo cual son mucho más rápidos
- Trabajan en capas y cada vez que agreguemos, modifiquemos o quitemos dependencias vamos a general una nueva versión de nuestro contenedor, es decir, una capa que irá por encima de la anterior
    - Esto se diferencia de un sistema de versionado como Git o Google Docs
- Para implementar un contenedor:
    - Lo primero que debemos hacer es crear una imagen
    - Solo se necesita encontrar una imagen base del software en donde montar el sistema

        - Las imágenes base que forman parte de nuestro build inicial (docker build) del contenedor pueden ser el SO, la base de datos y nuestra aplicación software
        - Luego debemos hacer un publish (docker publish) o publicar el contenedor para que este se suba al repositorio y que cualquier persona pueda utilizarla haciendo un pull (docker pull)
        - Por último, ejecutar el programa con un servicio más, haciendo run (docker run) al contenedor
    - Gracias a esto, las implementaciones de versiones que antes demoraban días se pueden realizar en segundos

Características:

- La mejor caracterísitica de contenedores es que podamos **configurar** el sistema fácilmente y también más rápido.
    - Los requisitos de la infraestructura ya no están vinculados con el entorno de la aplicación, ya que se puede utilizar en una amplia variedad de entornos
- El proporcionar una huella más pequeña del sistema operativo a través de contenedores, un contenedor tiene la capacidad de reducir el **tamaño** del desarrollo
- Utilizar contenedores equivale a aumentar la **productividad**
    - Esto facilita la configuración técnica y el despliegue rápido de la aplicación. Además, ayuda a ejecutar la aplicación en un entorno aislado y reduce los recursos
- Existen herramientas de programación y clustering para contenedores
    - Algunos contenedores exponen una web y otros ofrecen API como su front end, que nos permite utilizar varias herramientas para controlarlo
    - Nos ayuda a controlar un cluster de hosts contenedores como un único host virtual (**gestión múltiple**)
- La lista de tareas que nos permite especificar el estado del contenedor dentro de un cluster y los **servicios**
    - Cada tarea representa una instancia de un contenedor que debe estar en ejecución y que puede ser programada sobre los nodos (cada instancia que lo ejecuta)
- Los contenedores se utilizan para ejecutar aplicaciones en un entorno aislado (**isolado**)
    - Lo mejor de esta característica de los contenedores es que aquí cada contenedor es independiente de otro y, además, nos permite ejecutar cualquier tipo de aplicación requerida
- Los contenedores proporcionan configuraciones por defecto que ofrecer una mayor protección para las aplicaciones que se ejecutan sobre ellos y a través de orquestadores

    - La plataforma establece valores predeterminados **seguros**, al tiempo que deja los controles en manos del administrador para cambiar las configuraciones y las políticas según sea necesario

ADMINISTRADORES DE CONTENEDORES

Los **orquestadores** son sistemas de automatización del despliegue, ajusta de escala, manejo de aplicaciones (administración), comunicación y disponibilidad de nuestro software en **contenedores**. Contenedores y orquestadores deben ser incorporados conjuntamente
Hay entornos en los que necesitamos que no hayan tiempos de inactividad, por lo que si un contenedor se cae, otro debe iniciarse automáticamente. Lo mejor es que esto sea implementado por un orquestador

Características
Un orquestados se encarga de cosas como:

- Autorreparación
    - Puede recuperar los contenedores que fallen, ya sea que tenga que reemplazarlos o dar de baja a los que no respondan
- Retroceso automatizado
    - Capacidad de retroceder y así evitar tener que detener el sistema, hacer un backup y luego volver a iniciar el sistema
- Configuración automática
- Despliegue y "levantado" automático de servicios basados en contenedores
- Balanceado de carga
    - Cuando un contenedor recibe mucha demanda, el orquestador es capaz de distribuir el tráfico de red de manera que sea estable y balanceado
- Autoescalado y autoreinicio de contenedores
    - Cuando se producen picos de demanda se requieren muchos más recursos de computación de los servidores. Esta es una ventaja en la nube moderna, ya que los costos se basan en los recursos consumidos
- Control de la "salud" de cada contenedor
- Intercambio de datos y networking
- Mantenimiento de parámetros "secretos" y configuraciones

Ejemplos de orquestadores

- **Kubernetes**
    - Es el motor de orquestación de contenedores más popular que existe en el mercado.

    - Comenzó siendo un proyecto de Google. Actualmente, miles de equipos de desarrolladores lo usan para desplegar contenedores en producción.
    - La herramienta funciona agrupando contenedores que componen una aplicación en unidades lógicas para una fácil gestión y descubrimiento
- **Docker Swarm**
    - Swarm es la solución que propone Docker ante los problemas de los desarrolladores a la hora de orquestar y planificar contenedores a través de muchos servidores.
    - Viene incluido junto al motor de Docker y ofrece muchas funciones avanzadas integradas —como el descubrimiento de servicios, balanceo de carga, escalado y seguridad—
- **Mesosphere DC/OS**
    - El sistema operativo Mesosphere Datacenter (DC/OS) es una plataforma de código abierto, integrada para datos y contenedores desarrollados sobre el kernel de sistema distribuido Apache Mesos
    - Se ha diseñado para gestionar múltiples máquinas dentro de un centro de datos con uno o más clústeres, ya sea en la nube o usando software en servidores en local
    - DC/OS puede desplegar contenedores y gestionar tanto aplicaciones sin estado como protocolos con estado en el mismo entorno
    - Es capaz de funcionar con Docker Swarm y Kubernetes
- **HashiCorp Nomad**
    - Soportada por Linux, Mac y Windows, Nomad es una herramienta binaria única capaz de planificar todas las aplicaciones virtualizadas en contenedores o independientes
    - Nomad ayuda a mejorar la densidad, a la vez que reduce costos, ya que es capaz de distribuir de manera eficiente más aplicaciones en menos servidores
- **Amazon ECS**
    - El servicio de AWS es un sistema de gestión muy escalable que permite a los desarrolladores ejecutar aplicaciones en contenedores
    - Está formado por muchos componentes integrados que permiten la fácil planificación y despliegue de clústeres, tareas y servicios del contenedor
- **Amazon Elastic Kubernetes Service**
    - Amazon EKS facilita la implementación, la administración y el escalado de aplicaciones en contenedores mediante Kubernetes en AWS
    - Ejecuta la infraestructura de administración de Kubernetes por el usuario en varias zonas de disponibilidad de AWS para disminuir errores
    - Las aplicaciones que se ejecutan en cualquier entorno estándar de Kubernetes son totalmente compatibles y pueden migrar fácilmente a Amazon EKS
- **Azure Kubernetes Service (AKS)**

    - El servicio de Azure es código abierto y está optimizado para su uso en las máquinas virtuales de Azure, denominadas Azure Virtual Machines
    - Proporciona las herramientas necesarias para crear, configurar y gestionar la infraestructura de contenedores Docker abiertos
    - AKS ofrece desarrollo simplificado de aplicaciones basadas en contenedores y despliegue con soporte para Kubernetes, Mesosphere DC/OS o Swarm para la orquestación
- **Google Kubernetes Engine (GKE)**
    - Montado sobre Kubernetes, permite desplegar, gestionar y escalar aplicaciones de contenedores en la nube de Google
    - El objetivo de GKE es optimizar la productividad del departamento de desarrollo al mejorar la gestión de las cargas de trabajo basadas en contenedores
    - Oculta tanto las tareas de gestión simple como aquellas más complejas detrás de herramientas de líneas de comando, usando interfaces transparentes y fáciles de usar
    - Obviamente, Kubernetes es la columna vertebral de GKE
    - Aunque no es estrictamente necesario dominar Kubernetes para usar GKE, ayuda mucho si al men