**II\_11\_MAQUINAS\_VIRTUALES**

**ESCRITORIOS REMOTOS**

Programas que nos permiten interactuar con una computadora a través de internet.

La computadora remota HOST y la compu desde la que se trabaja CLIENTE.

VENTAJA: Ahorro de recursos.

DESVENTAJAS: si el programa que brinda el escritorio remoto, puede recibir ciberataques. Depende de la calidad de la conexión de internet la conexión al escritorio remoto.

**MAQUINAS VIRTUALES**

Software capaz de contener un SO en su interior. Haciéndole creer que es una compu de verdad. Ese SO puede albergar otro más.

Tipos:

SISTEMAS: emula compu completa. Otro SO en su interior. Lugar donde es creada Hipervisor. 2 tipos: 1 rápido y seguro, es el más utilizado. Tipo 2, corre sobre SO y es más lento q el anterior.

PROCESOS: emula sólo un proceso completo, x ejemplo una aplicación. Sirve para desarrollar aplicaciones que se ejecutaran en otros sistemas operativos. Ventaja: ejecutar aplicaciones de otros SO. Entorno de seguridad.

En serivodres mejora el aprovechamiento del hardware.

Ventajas: almacenamiento y redes.

DESVENTAJAS: menos eficientes q las máquinas reales, acceden al hardware de forma indirecta, ralentizando el proceso. Cuando varias máquinas virtuales se ejecutan en la misma, los recursos pueden verse afectados.

**CONTENEDORES**

Software que desarrollamos, SO y librerías 🡪 todo se empaqueta en un contenedor.

Los C funcionan de manera conjunta con nuestro SO x lo q no requieren un hipervisor y trabajan más rápido.

Trabajan en capas.

Para implementar un contendeor necesitamos crear una imagen. Una imagen base para el contendeor, luego hacer un docker build (SO, BD, software), un docker publish (xa subirlo al repositorio) , docker pull y docker run (xa acceder y ejecutarlo)

Un contenedor es un concepto de

empaquetación de software que

incluye la aplicación y todas sus

dependencias de ejecución.

**La configuración**

La mejor característica de contenedores es

que podemos configurar el sistema

fácilmente y también más rápido. Es posible

desplegar nuestro código en menos tiempo

y esfuerzo con la ayuda de contenedores.

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 tamaño**

Al proporcionar una huella más pequeña del sistema operativo a través de

contenedores, un contenedor tiene la capacidad de reducir el tamaño del

desarrollo.

**La productividad**

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.

**Gestión múltiple**

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. Además, nos ayuda

a controlar un clúster de hosts

contenedores como un único

host virtual.

**Los servicios**

La lista de tareas que nos permite

especificar el estado del contenedor

dentro de un cluster y los servicios.

Básicamente, 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).

**La isolación**

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.

**La seguridad**

Los contenedores proporciona

configuraciones por defecto que

ofrecen 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.

**ORQUESTADORES**

**Los orquestadores son sistemas de**

**automatización del despliegue, ajuste**

**de escala y manejo de aplicaciones en**

**contenedores.**

Características Un orquestador de contenedores se ocupa de cuestiones como: ● Configuración automática. ● Despliegue y "levantado" automático de servicios basados en contenedores. ● Balanceado de carga. ● Autoescalado y autoreinicio de contenedores. ● Control de la "salud" de cada contenedor. ● Intercambio de datos y networking. ● Mantenimiento de parámetros "secretos" y configuraciones.

**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 menos conocemos sus fundamentos básicos.

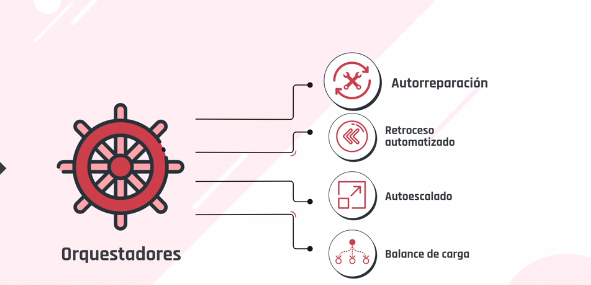
**HERRAMIENTA Q AUTOMATIZA:**

****

**EJECUTÁNDOSE EN CONTENEDORES.**

**HAY ENTORNOS EN LOS Q NO SE ´PUEDEN PERMITIR TIEMPOS DE INACTIVIDAD POR LO QUE SI UN CONTENEDOR SE CAE DEBE EJECUTARSE OTRO.**

**CARACTERISTICAS**

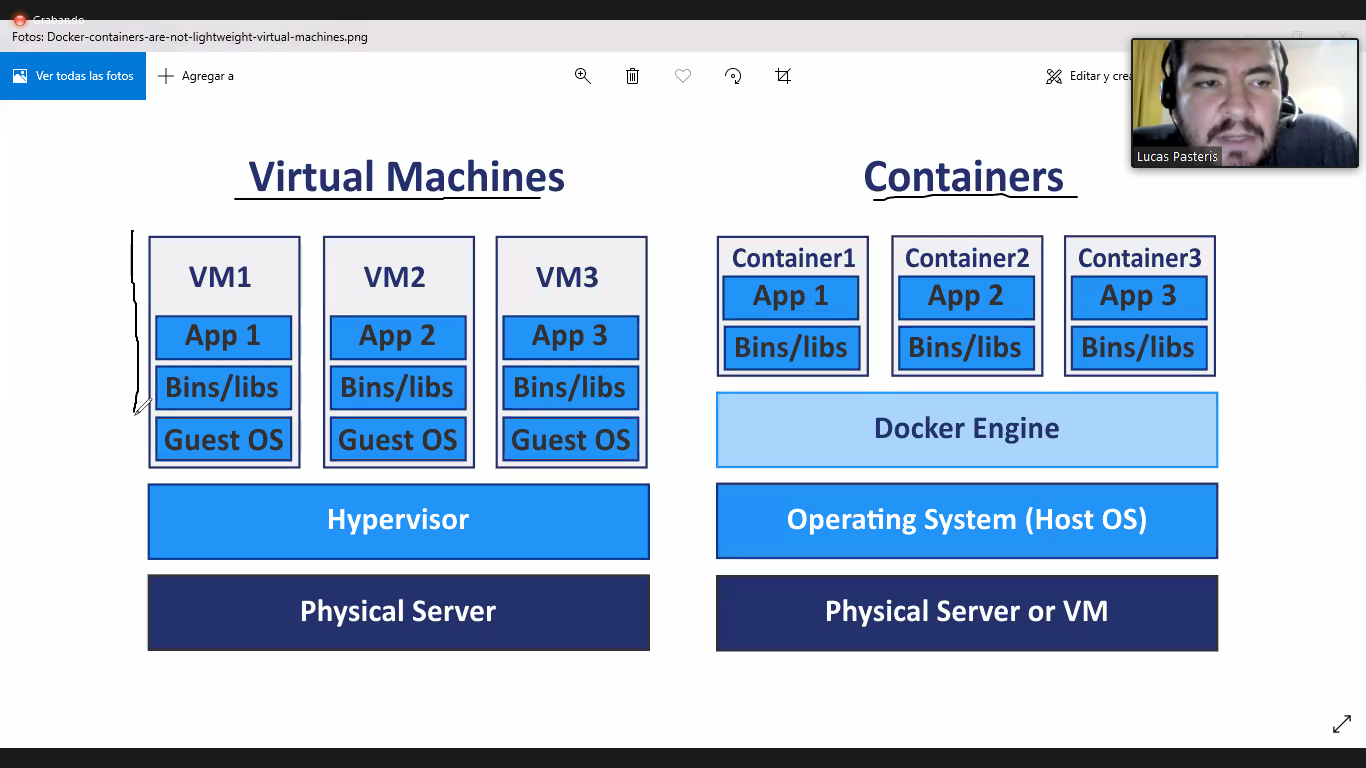
****

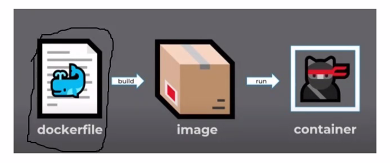
****

**BALANCE DE CARGA**

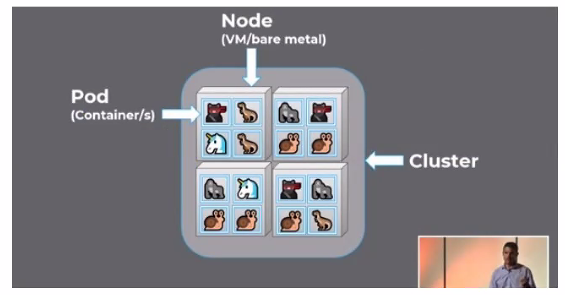
****

**Clase 19/04/21**



**DOCKER**

**Orquestador:**

****

Para aprender sobre contenedores:

**https://www.katacoda.com/courses/container-runtimes**