**Ensayo de Arquitectura**

Gabriel Silva 29803338

Juan Sanabria 30276906

**Virtualización**

La virtualización es un proceso que permite crear versiones virtuales de recursos computacionales, como sistemas operativos, almacenamiento o redes, en lugar de depender de hardware físico. Se utiliza para optimizar la eficiencia, facilitar la gestión de recursos y ejecutar múltiples entornos en un solo servidor físico.

**Tipos de virtualización:**

*Virtualización de Hardware:*

Permite ejecutar múltiples sistemas operativos (máquinas virtuales) sobre una única máquina física.

Puede ser usado en un servidor físico que aloje varias maquinas virtuales donde cada una ejecute un sistema operativo diferente,para así optimizar recursos.

*Virtualización de Red:*

Abstrae los recursos de red para crear redes virtuales aisladas.

Se puede usar en entornos de desarrollo y pruebas,creando redes virtuales para proyectos diferentes,permitiendo simular diferentes escenarios de red.

*Virtualización de Almacenamiento:*

Separa el almacenamiento físico del acceso lógico, mejorando la eficiencia y flexibilidad.

Utilizado en entornos empresariales para implementar almacenamiento compartido, facilitando la administración y el escalado.

*Virtualización de Escritorio:*

Proporciona escritorios virtuales accesibles desde cualquier dispositivo.

Puede ser usado en una empresa que implemente dichos escritorios virtuales para que sus empleados accedan desde cualquier lugar,mejorando la movilidad.

*Virtualización de Sistema Operativo (Contenedores):*

Utiliza contenedores para encapsular aplicaciones y sus dependencias, compartiendo el núcleo del sistema operativo.

Se puede hacer un despliegue de aplicaciones en entornos de desarrollo, pruebas y producción de manera consistente y eficiente.

**Herramientas de Virtualización:**

***Virtualización de Hardware.***

***VirtualBox****:*

Usado para desarrollo, pruebas y ejecución de sistemas operativos en entornos virtuales. Útil para crear laboratorios de pruebas.

***Vmware****:*

Se usa para la implementación de entornos de servidores virtuales en entornos empresariales. Permite la consolidación de servidores físicos.

***Xen:***

Usado para virtualización de servidores para mejorar la eficiencia de recursos y permitir la migración de máquinas virtuales entre servidores físicos.

***QEMU:***

Sirve para la emulación de arquitecturas de hardware diferentes, lo que permite la ejecución de sistemas operativos y programas diseñados para plataformas específicas.

*Docker:*

Virtualización de Sistema Operativo (Contenedores).

Se usa en empaquetamiento y distribución de aplicaciones junto con sus dependencias. Facilita la implementación y escalabilidad de aplicaciones en entornos de contenedores.

*Vagrant:*

Gestor de Entornos de Desarrollo (No es una herramienta de virtualización, pero se utiliza comúnmente con máquinas virtuales).

Vagrant es una herramienta de código abierto que ayuda a gestionar entornos de desarrollo reproducibles. Aunque no realiza virtualización directamente, trabaja con proveedores de virtualización (como VirtualBox, VMware, etc.) para crear y configurar entornos de desarrollo consistentes.

**MPI (Message Passing Interface) o (Interfaz de paso de mensajes)**

MPI es un estándar de comunicación utilizado para la programación paralela en sistemas distribuidos. Permite la comunicación entre procesos que se ejecutan en nodos separados de un clúster.

*Características Principales:*

Paso de Mensajes: La comunicación entre procesos se realiza mediante el intercambio explícito de mensajes.

Escalabilidad: Permite diseñar aplicaciones paralelas que pueden escalar para aprovechar eficientemente recursos en sistemas distribuidos.

Portabilidad: Al ser un estándar, las implementaciones de MPI están disponibles en varias plataformas, haciendo que el código sea portable.

*Beneficios:*

Permite desarrollar aplicaciones paralelas en entornos distribuidos.

Aprovecha eficientemente recursos de clústeres.

Mejora el rendimiento de cálculos intensivos distribuyendo la carga de trabajo.

*Limitaciones:*

Requiere un clúster o sistema distribuido para aprovechar completamente sus capacidades.

El diseño y la implementación de algoritmos paralelos pueden ser más complejos que en modelos de programación secuenciales por lo que MPI es una herramienta poderosa para programadores que buscan aprovechar el paralelismo en entornos distribuidos.

**OpenMP (Open Multi-Processing):**

OpenMP es una API de programación que facilita la creación de programas paralelos en sistemas de memoria compartida. Permite a los desarrolladores agregar directivas de compilador a su código para indicar áreas que se pueden ejecutar en paralelo.

*Características Principales:*

Memoria Compartida: OpenMP se centra en la programación paralela en sistemas de memoria compartida, donde varios hilos pueden acceder a la misma región de memoria.

Directivas de Compilador: Se utilizan directivas de compilador para indicar regiones de código que pueden ejecutarse en paralelo.

Simplicidad: Comparado con MPI, OpenMP es más simple de implementar y se integra con el código existente de manera más directa.

*Beneficios:*

Simplifica la creación de programas paralelos en sistemas de memoria compartida así aprovecha la concurrencia mediante hilos en un solo procesador.

Integración más sencilla en comparación con modelos de programación distribuida.

*Limitaciones:*

Se limita a sistemas de memoria compartida, y puede no escalar tan bien en sistemas distribuidos.

La competencia por recursos compartidos puede introducir condiciones de carrera, lo que requiere precauciones.

OpenMP es una opción excelente para programadores que desean paralelizar tareas en sistemas de memoria compartida. Es más sencillo de implementar que modelos de programación distribuida como MPI.

**CUDA (Compute Unified Device Architecture):**

CUDA es una plataforma de computación paralela y un modelo de programación desarrollado por NVIDIA. Permite a los desarrolladores utilizar las GPU de NVIDIA para realizar cálculos de propósito general, no solo gráficos. CUDA se basa en la arquitectura de la GPU y proporciona un conjunto de extensiones para el lenguaje de programación C.

*Características Principales:*

GPU como Coprocesador: Permite a los desarrolladores utilizar la potencia de cálculo de la GPU junto con la CPU para acelerar tareas computacionales intensivas.

Modelo de Programación: Utiliza extensiones de C para definir funciones que se ejecutan en la GPU. Estas funciones se conocen como “kernels” y se ejecutan en paralelo en muchos núcleos de la GPU.

Jerarquía de Hilos y Bloques: CUDA organiza la ejecución en hilos y bloques, lo que facilita la explotación de la paralelización masiva.

*Beneficios:*

Aprovecha la potencia de cálculo masiva de las GPU.

Permite la ejecución de cientos o miles de hilos en paralelo.

Puede acelerar significativamente tareas computacionales intensivas.

*Limitaciones:*

Depende de la disponibilidad de hardware compatible con CUDA (GPU de NVIDIA).

No es una solución universal para todas las tareas. CUDA es una herramienta poderosa para la computación paralela en GPU, especialmente en entornos donde se dispone de hardware NVIDIA compatible.

**RAID (Redundant Array of Independent Disks) Matriz redundante de discos independientes:**

RAID es una tecnología de almacenamiento que utiliza múltiples discos duros para mejorar el rendimiento, la confiabilidad o ambas cosas, según la configuración. Los discos se agrupan en una sola unidad lógica, y los datos se distribuyen o replican entre ellos. Cada configuración RAID se denomina nivel de RAID, y cada nivel tiene sus propias características y aplicaciones específicas.

*RAID 0:*

Divide los datos en bloques y los distribuye a través de múltiples discos. No ofrece redundancia ni espejo. Ofrece mayor rendimiento en lectura y escritura debido a la distribución de datos entre discos.

La pérdida de un disco implica la pérdida de todos los datos del conjunto.

Se utiliza en situaciones donde se busca mejorar el rendimiento sin importar la redundancia, como en aplicaciones que requieren una alta velocidad de transferencia de datos.

*RAID 1:*

Crea una copia exacta (espejo) de los datos en dos discos diferentes.

Si un disco falla, los datos aún están disponibles en el espejo.

Rendimiento similar al de un solo disco, ya que todas las operaciones de lectura se pueden realizar desde cualquiera de los discos en el conjunto.

Es utilizado en entornos donde la redundancia y la disponibilidad de datos son críticas, pero el rendimiento no es una prioridad.

*RAID 5:*

Distribuye datos y paridad (información de verificación de errores) a través de varios discos. Ofrece un equilibrio entre rendimiento y redundancia, tiene mayor rendimiento que RAID 1.

El conjunto puede resistir la pérdida de un disco.

Es adecuado para entornos empresariales donde se requiere un buen rendimiento y una capacidad de recuperación ante fallos.

*RAID 10:*

Combina características de RAID 1 y RAID 0. Esencialmente, es un conjunto de discos RAID 1 configurados en un conjunto RAID 0.

Tiene mayor rendimiento en comparación con RAID 1, alta redundancia y resistencia a fallos de disco.

Es ideal para entornos empresariales que requieren un equilibrio entre rendimiento y redundancia crítica, como servidores que manejan bases de datos importantes.