RESUMEN: Máquinas virtuales (Parte 1 y Parte 2)

Introducción

El material de clase presenta el concepto de virtualización aplicada a sistemas operativos y a la administración de infraestructura. La idea central es separar el software del hardware físico para ejecutar varios sistemas "aislados" sobre el mismo equipo, aumentando el aprovechamiento de recursos y la flexibilidad operativa. A lo largo de las dos partes se recorren las piezas básicas (hipervisores, máquinas virtuales y redes/almacenamiento virtuales), las operaciones frecuentes de laboratorio (creación, clonación, snapshots, migración), así como beneficios, límites y buenas prácticas de seguridad.

1) ¿Qué es la virtualización?

Virtualizar significa ofrecer a un sistema invitado (guest) la ilusión de tener un computador propio. Para lograrlo, una capa de software llamada hipervisor intercepta y media el acceso a CPU, memoria, almacenamiento y red del equipo anfitrión (host). Gracias a ello, es posible consolidar varios servidores lógicos en una sola máquina física, aislar entornos de prueba y simplificar la recuperación ante fallos.

2) Arquitectura general e hipervisores

Existen dos enfoques arquitectónicos muy difundidos. En los hipervisores de Tipo 1 (bare-metal), el hipervisor corre directamente sobre el hardware y encima de él se ejecutan las VMs; es típico de centros de datos por su desempeño y control fino de recursos (ej.: ESXi, Hyper-V Server, Xen). En los de Tipo 2 (hosted), la virtualización es una aplicación que corre sobre un sistema operativo anfitrión (ej.: VirtualBox, VMware Workstation); resulta práctico para laboratorios, docencia y equipos personales. En ambos casos, el hipervisor expone dispositivos virtuales (vCPU, vRAM, vNIC, discos virtuales) y administra acceso concurrente al hardware real.

3) Modalidades de virtualización

Tres ideas aparecen de forma recurrente en la literatura y en la práctica:

• Virtualización completa: el invitado cree hablar con hardware real; las instrucciones privilegiadas se manejan por el hipervisor. Fácil para el usuario, pero con cierto costo de rendimiento.

• Paravirtualización: el invitado coopera con el hipervisor mediante controladores y "llamadas hiper" (ej.: controladores virtio). Gana eficiencia en I/O a cambio de requerir componentes específicos.

• Asistencia por hardware: extensiones como Intel VT-x/VT-d y AMD-V permiten ejecutar instrucciones sensibles de modo seguro, reduciendo la traducción/binario y mejorando el rendimiento.

4) Gestión de recursos en una VM

CPU. El planificador del hipervisor distribuye tiempo de CPU entre VMs. Es común el "overcommit" (asignar más vCPU lógicas que núcleos físicos) cuando las cargas no usan todo el tiempo de CPU a la vez; mal usado, causa latencias y "robo de CPU".

Memoria. Además de la asignación estática, aparecen técnicas como ballooning (devolver

memoria no usada al host) y deduplicación de páginas (KSM) para ahorrar RAM. Si el host entra en swap, todas las VMs sufren.
Almacenamiento. Los discos virtuales suelen ser archivos en el host (VDI/VMDK/QCOW2). El aprovisionamiento "thin" empieza pequeño y crece bajo demanda; el "thick" reserva espacio desde el inicio. Los snapshots guardan el estado en un punto en el tiempo: útiles para laboratorio, pero muchos snapshots degradan el rendimiento.
Red. Las VMs se conectan a switches virtuales. Los modos típicos son NAT (salida a internet sin exponer puertos), bridge (la VM aparece en la red como otro equipo más) y host-only (red privada entre host y VMs).

5) Operaciones habituales que muestra el curso (laboratorio)
• Crear y configurar una VM: elegir SO invitado, núcleos y RAM razonables, disco virtual, adaptador de red y controladores paravirtuales para buen I/O.
• Clonar / crear plantillas: acelera la réplica de entornos homogéneos para prácticas y pruebas.
• Snapshots: tomar un "punto seguro" previo a un cambio riesgoso y revertir si algo falla. No son reemplazo de backups completos.
• Migración (cuando la plataforma lo soporta): mover una VM encendida entre hosts físicos para mantenimiento o balanceo (p. ej., vMotion/Live Migration). Requiere almacenamiento compartido y red de baja latencia.

6) Beneficios y limitaciones
Beneficios: consolidación de servidores (mejor uso de hardware), despliegues rápidos, entornos de prueba aislados, recuperación más sencilla (restaurar imágenes/snapshots), y mejor continuidad del negocio gracias a la migración y a la orquestación.
Limitaciones: sobrecarga (overhead) frente a ejecución nativa, contención de recursos en picos, complejidad operativa (capas adicionales para diagnosticar problemas) y dependencia del hipervisor. La latencia de I/O y el rendimiento gráfico 3D pueden ser desafiantes según el caso de uso.

7) Seguridad en entornos virtualizados
El aislamiento entre VMs eleva la seguridad respecto a instalar todo "en el mismo host", pero no es perfecto. Buenas prácticas vistas en el curso: mantener el hipervisor actualizado, minimizar superficie de ataque (sólo servicios necesarios), usar redes separadas o VLANs para tráfico de administración, activar cifrado de discos virtuales cuando aplique, y aplicar microsegmentación/ACLs entre VMs para reducir movimientos laterales. Los "VM escape" son raros pero posibles, por lo que el parcheo oportuno es crítico.

8) Máquinas virtuales vs contenedores (comparación breve)
Las VMs virtualizan hardware y alojan un SO completo por instancia; arrancan más lento, pero ofrecen fuerte aislamiento de kernel y compatibilidad casi total. Los contenedores comparten kernel con el host, son más livianos y arrancan en segundos; optimizan densidad y portabilidad de aplicaciones, pero dependen de un kernel común y requieren controles de seguridad (namespaces, cgroups, perfiles AppArmor/SELinux). En prácticas docentes, las VMs resultan ideales para emular topologías, experimentar con distintos SO y crear

escenarios “rompibles” sin afectar el host; los contenedores destacan para empaquetar y reproducir servicios.

9) Recomendaciones prácticas para el laboratorio
• Dimensiona con cabeza fría: asigna sólo los recursos que de verdad necesitas y observa el consumo real antes de crecer.
• Prefiere controladores paravirtualizados (virtio) y discos “thin” para ahorrar espacio, usando snapshots con moderación.
• Documenta cambios y toma snapshots antes de experimentar; borra los que ya no necesites.
• Separa la red de administración de la red de datos de las VMs; evita exponer servicios innecesarios.
• Automatiza lo repetitivo: plantillas, cloud-init/Vagrant y scripts de aprovisionamiento acortan tiempos.

Conclusiones
La virtualización es una palanca transversal en Sistemas Operativos II: permite aprender con entornos realistas y, al mismo tiempo, enseña a administrar recursos de forma responsable. Entender cómo decide el hipervisor (planificación, memoria, I/O), cómo se conectan las VMs a la red y qué implicaciones tienen snapshots y migraciones ayuda a construir laboratorios sólidos, reproducibles y seguros. Con esos fundamentos, elegir entre VMs o contenedores deja de ser una cuestión de “moda” y pasa a ser una decisión técnica según aislamiento, rendimiento, compatibilidad y rapidez de despliegue.

Referencias (fuente del contenido de clase)
• Video: Cap 14 “Máquinas virtuales – Parte 1” (YouTube)
• Video: Cap 14 “Máquinas virtuales – Parte 2” (YouTube)