First of all they differ in types, e.g. JVM is Application virtual machine, when mostly one process is executing, organized syscall virtualization and application code translation. For the other it lies on the hardware virtualization level.

VMware, since it builds cloud solutions, has a compound architecture. On the top level overview it consists of real hardware, ESX Servers, VMFS (virtual machine file system), Virtual SMP (virtual symmetric multi-processing) – a single virtual machine can use several physical processors, virtual machines themselves and VirtualCenter Management Server. The *hosted* approach – when partitioning provided with services on top of a standard operating system and supports the broadest range of hardware configurations, and *hypervisor* approach – is the first layer of software installed on a clean x86-based system. Since it has direct access to the hardware resources, a hypervisor is more efficient than hosted architectures, enabling greater scalability, robustness and performance. VMware cloud infrastructure runs everything on hardware, using VirtualCenter to manage everything, and has ESX servers with a hypervisor that provides: cpu, mmu, i/o virtualization. Next level of ESX server is the part of VMFS, Virtual Networking, Resource management and so on, and surely the last one is VMM (virtual machine monitor). Also they have distributed services tied to VirtualCenter as VMotion, Provisioning, DRS (distributed resource scheduler), Consolidation Backup. They use *para-virtualization* techniques, when operating system compatibility is traded off against performance for certain CPU-bound applications running on systems without virtualization hardware assist. Of course, these are modern cloud solutions, whereas VMware workstation uses *hosted* architecture.

By the way, Hyper-V is a first type hypervisor, whereas VMware workstation uses second type. But for infrastructure it's very similar to the VMware solution. It also uses partitioning, servers and virtualization service providers. Hyper-V uses hardware-assisted virtualization, such as Intel VT-x and AMD-V. Hyper-V uses a technique called Dynamic Memory (ref. to "ballooning" technique), this allocates memory to VMs as needed.

Xen uses VDI – a variant of the client-server model of desktop virtualization, which uses host-based virtual machines. XenDesktop also uses e.g. ballooning technique. As Hyper-V it also uses Intel VT-x and AMD-V hardware-assisted virtualization. Overall it could be said that XenDesktop solution for the user is as follows: there are Store Front and Delivery Controller, that communicate with every

Virtual Delivery Agent (installed on server or workstation operating systems, the VDA enables connections for desktops and apps. For remote PC access, install the VDA on the office PC), where one of them has a hypervisor and virtual machine, and for the rest is some special services through Provisioning Services. It also lies on VMware vSphere.

Qemu is very interesting, but it could work slowly on the old machines because of the absence of VT-x or AMD-V. As for many of them it also uses ballooning, memory mapping. In the KVM architecture, the virtual machine is implemented as a regular Linux process, scheduled by the standard Linux scheduler. In fact, each virtual CPU appears as a regular Linux process. This allows KVM to benefit from all the features of the Linux kernel. KVM inherits powerful memory management features from Linux. The memory of a virtual machine is stored the same as memory is for any other Linux process and can be swapped, backed by large pages for better performance, shared, or backed by a disk file. NUMA support (Non-Uniform Memory Access, memory design for multiprocessors) allows virtual machines to efficiently access large amounts of memory. KVM supports memory virtualization features such as Intel's Extended Page Table (EPT) and AMD's Rapid Virtualization Indexing (RVI). KVM hypervisor uses the VirtIO standard developed by IBM and Red Hat in conjunction with the Linux community for paravirtualized drivers;

For some of them it could use Ring 0 level since it has hardware support, but sometimes due to unavailability it is forced to use Ring 1 (which will be slower).

References

1. Paravirtualization, https://pediaa.com/what-is-the-difference-between-full-virtualization-and-paravirtualization-in-cloud/
2. VMware Virtualization Overview, https://www.vmware.com/pdf/virtualization.pdf
3. VMware Architecture Infrastructure Overview, https://www.vmware.com/pdf/vi_architecture_wp.pdf
4. The Architecture of VMware ESXi, https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/ESXi_architecture.pdf
5. VMware Workstation Pro, https://www.techtarget.com/searchvmware/definition/VMware-Workstation
6. Type 1 and Type 2 hypervisors, https://www.techtarget.com/searchitoperations/tip/Whats-the-difference-between-Type-1-vs-Type-2-hypervisor
7. Compare VMware Workstation Pro vs. Microsoft Hyper-V, https://www.techtarget.com/searchvmware/tip/Compare-VMware-Workstation-Pro-vs-Microsoft-Hyper-V
8. VMware Workstation Virtualizing I/O devices, https://www.usenix.org/legacy/publications/library/proceedings/usenix01/sugerman/sugerman_html/node4.html
9. Hyper-V architecture, https://learn.microsoft.com/en-us/windows-server/administration/performance-tuning/role/hyper-v-server/architecture
10. Hyper-V terminology, https://learn.microsoft.com/en-us/windows-server/administration/performance-tuning/role/hyper-v-server/terminology
11. Hyper-V Dynamic Memory Overview, https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/hh831766(v=ws.11)
12. Configuring VM Memory, https://docs.citrix.com/en-us/xencenter/current-release/vms-memory.html

13. What is desktop virtualization?, https://www.citrix.com/solutions/vdi-and-daas/what-is-desktop-virtualization.html
14. Creating Virtual Desktop Infrastructure Using Xen Desktop, https://repository.stcloudstate.edu/cgi/viewcontent.cgi?article=1112&context=msia_etds
15. Common Principles of working QEMU KVM (RU), https://habr.com/ru/articles/466549/
16. KVM/QEMU Memory Ballooning, https://listman.redhat.com/archives/libvirt-users/2020-September/012588.html
17. IBM: Dive into the KVM hypervisor, https://developer.ibm.com/articles/cl-hypervisorcompare-kvm/
18. IBM: Hypervisors, https://www.ibm.com/topics/hypervisors
19. IBM: VMware, https://www.ibm.com/topics/vmware

Author: Artem Chernitsa, B20-AI-01, a.chernitsa@innopolis.university