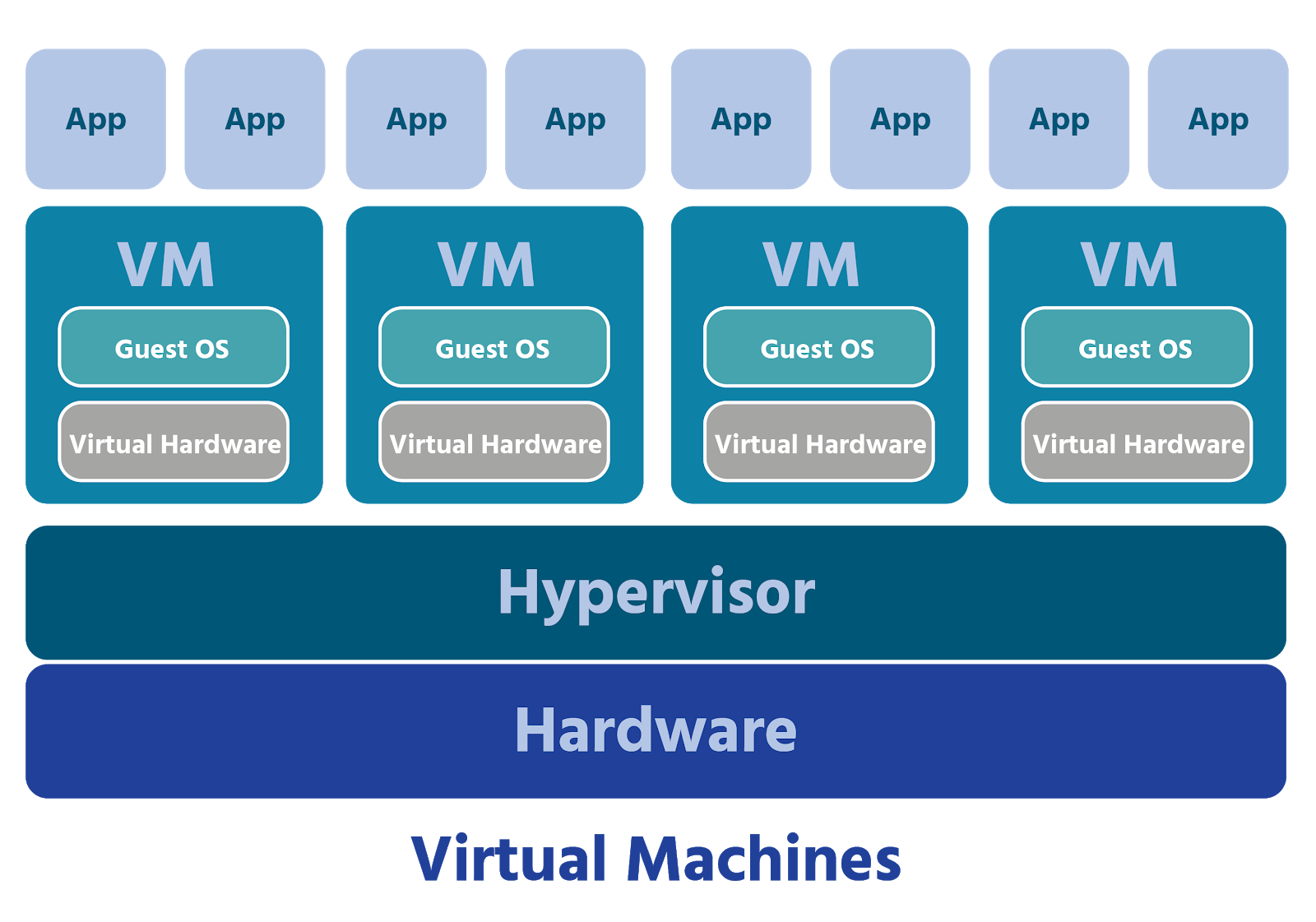
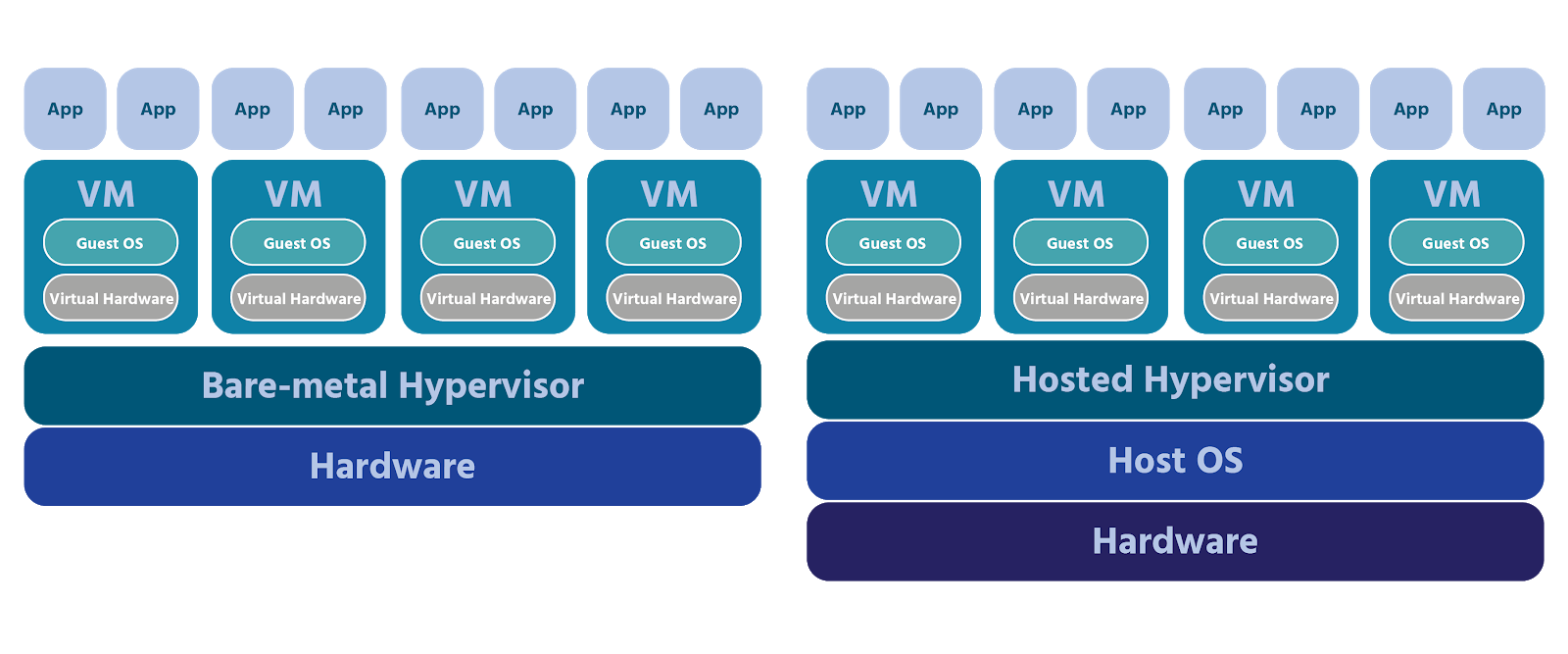
**26.2 Virtual Machines and Containers**

The first type of virtualized system is a *virtual machine*, or VM. This is where a complete operating system is installed on virtual hardware using a *hypervisor* to manage processes. The hypervisor is software that is installed on top of hardware, such as a server, creating a virtualization *layer* and acting as a platform for VMs to be created on. The hypervisor creates virtual resources such as CPUs, memory, hard disks and I/O communication interfaces like network and serial ports.

The first type of virtualized system is a *virtual machine*, or VM. This is where a complete operating system is installed on virtual hardware using a *hypervisor* to manage processes. The hypervisor is software that is installed on top of hardware, such as a server, creating a virtualization *layer* and acting as a platform for VMs to be created on. The hypervisor creates virtual resources such as CPUs, memory, hard disks and I/O communication interfaces like network and serial ports.



There are two methods of installing virtualization on a computer system. The first method is to install a *Type 1 hypervisor,* also called a bare-metal hypervisor, directly on top of the hardware. The Type 1 hypervisor sits directly between the hardware and the virtual machine. The second method is called *hosted virtualization*. To set up a system with hosted virtualization, a *Type 2 hypervisor*, also called a hosted hypervisor, needs to be installed on top of the operating system that already exists, the host operating system, instead of on the hardware like a bare metal hypervisor.

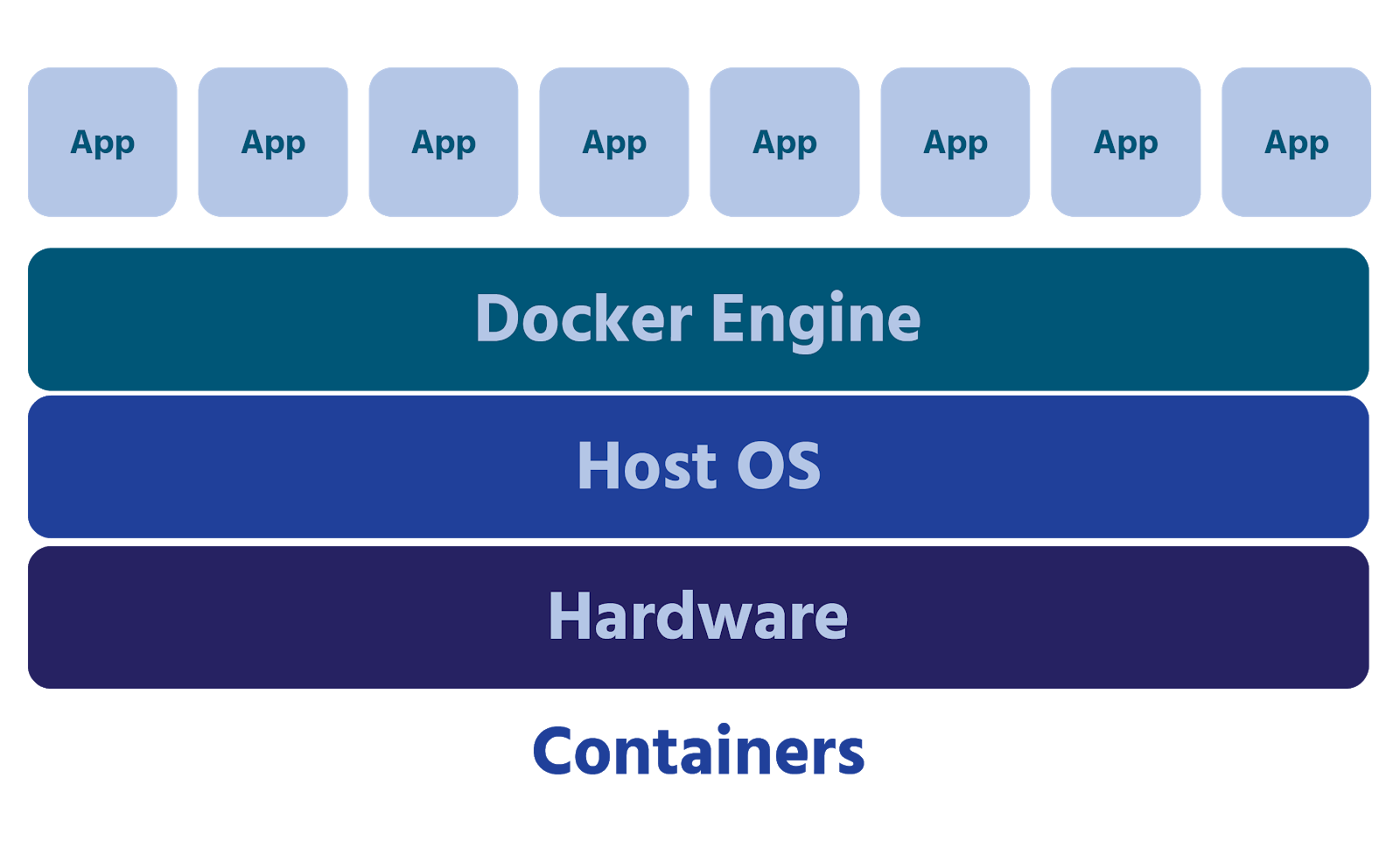


The operating system running in a VM is called a guest operating system. This operating system is usually installed as a pre-configured image, but it can also be built from installation sources such as CD, DVD, or PXE, to perform whatever tasks are needed by users. This configuration, along with installing application software, security patches, and network set-up is done by a systems administrator just as if it was a physical machine in a data center. Of course, being virtual, the VM might be sharing hardware with a number of other VMs. It might also be halfway around the world, physically closer to the clients that will be connecting to it.

**Consider This**

One of the main benefits of virtualization is that it makes computing resources more efficient. Traditionally, hardware could only run one computing system at a time, which may not have been a problem for individual users; however, with the growth of digital technology and increased needs for connectivity and data, data centers began adopting virtualization as a solution for providing computing resources efficiently.

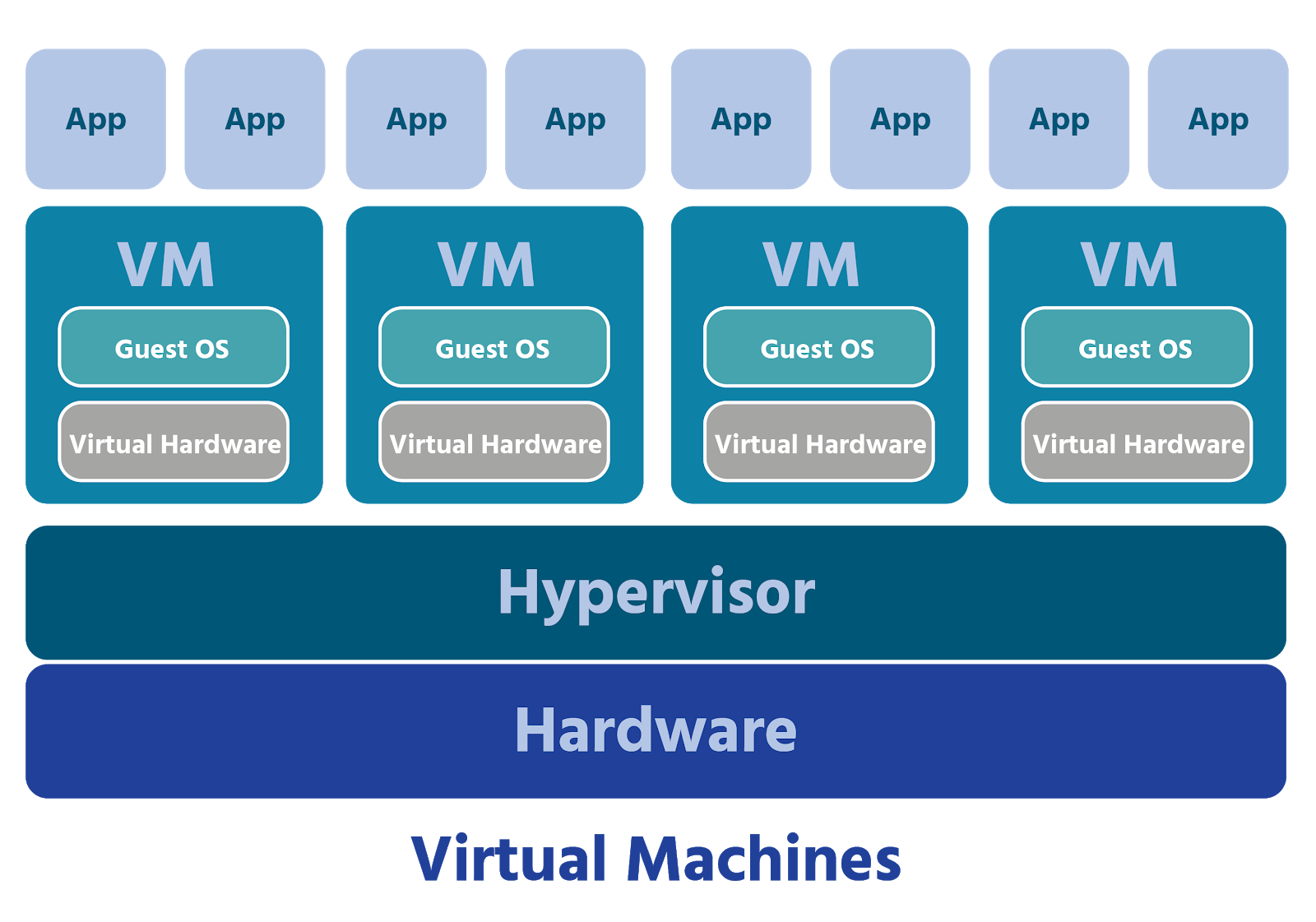
An outgrowth of virtualization, container technology is rapidly being adopted by software development organizations. *Linux Containers* allow system designers to bypass traditional operating systems and access computing resources differently. A containerized application relies on a container engine to communicate with the host OS without a hypervisor, or a guest VM. When programmers design these applications, they include all the dependencies needed to perform a specific task in the container. Also, typically, several containers are built which communicate with each other to perform tasks which previously would have been done by one program running on top of an OS.



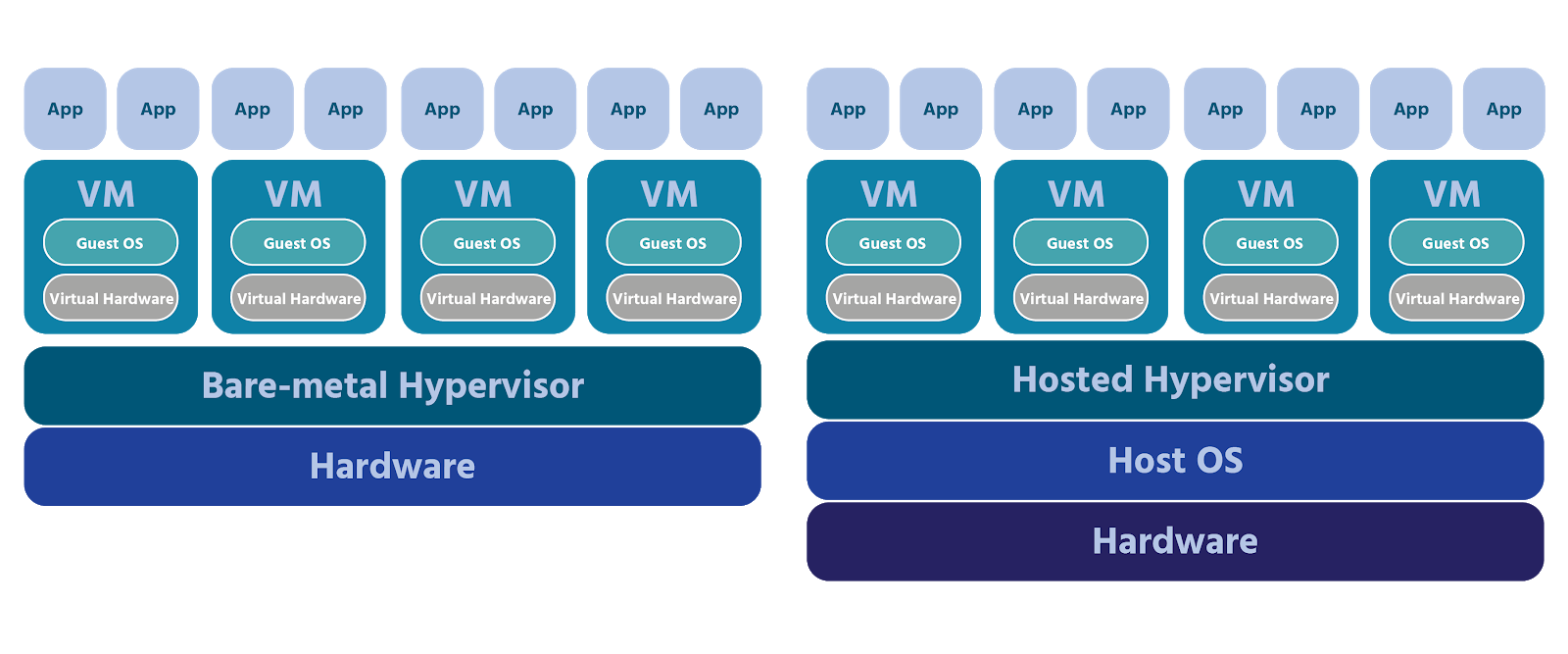
Doing this provides several advantages, including requiring fewer resources, cutting release cycle times, and abstracting program elements from host operating systems for better portability. Recently, there has been a move to Hybrid Container architecture. This consists of a multi-layered approach where a hypervisor, running on physical hardware, supports virtual machines that, in turn, host containers.

Organizations are weighing the pros and cons of the various configurations available today. Pros include: containers use fewer physical resources than VMs, offer enhanced reliability, and scalability. They also require less storage due to their lightweight nature and offer performance advantages both in startup time and application throughput.

On the con side, when containers are run directly on hardware, they can require considerable effort to ensure segregation and maintain security. When running on shared resources, the container engine needs to be tuned to avoid interference between and within containers. The engine also needs to control access to resources such as memory, I/O devices, and network connections. Finally, when programmers are allowed to create these small, stand-alone resources, it becomes easier for their numbers to get out of control. The advantages of containerization can quickly be outweighed by time and effort spent managing and maintaining them if discipline isn’t exercised from the beginning. Fortunately, the open source community has embraced container development and the tools for designing, deploying, and maintaining them continue to grow at a rapid pace.



There are two methods of installing virtualization on a computer system. The first method is to install a *Type 1 hypervisor,* also called a bare-metal hypervisor, directly on top of the hardware. The Type 1 hypervisor sits directly between the hardware and the virtual machine. The second method is called *hosted virtualization*. To set up a system with hosted virtualization, a *Type 2 hypervisor*, also called a hosted hypervisor, needs to be installed on top of the operating system that already exists, the host operating system, instead of on the hardware like a bare metal hypervisor.

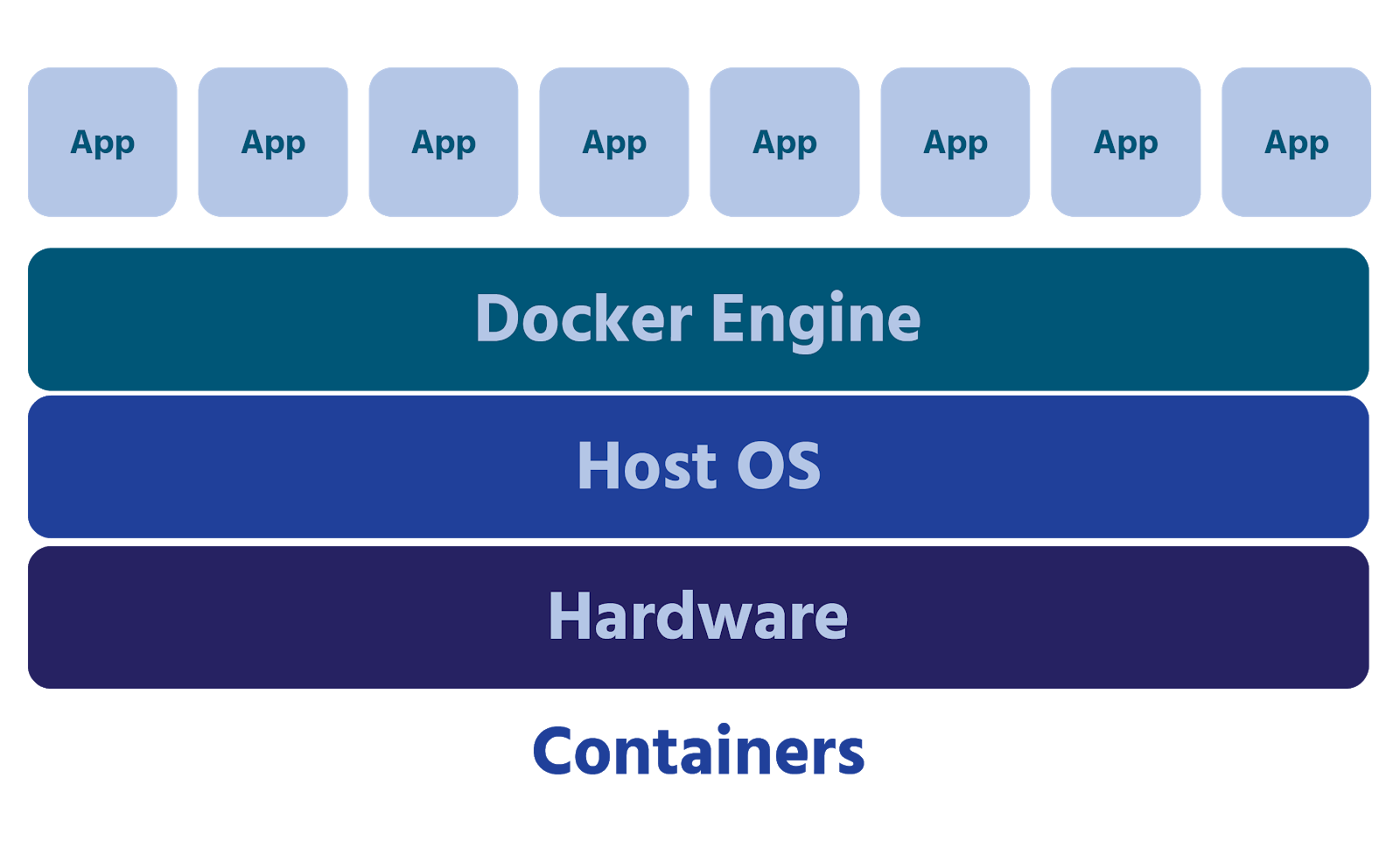


The operating system running in a VM is called a guest operating system. This operating system is usually installed as a pre-configured image, but it can also be built from installation sources such as CD, DVD, or PXE, to perform whatever tasks are needed by users. This configuration, along with installing application software, security patches, and network set-up is done by a systems administrator just as if it was a physical machine in a data center. Of course, being virtual, the VM might be sharing hardware with a number of other VMs. It might also be halfway around the world, physically closer to the clients that will be connecting to it.

**Consider This**

One of the main benefits of virtualization is that it makes computing resources more efficient. Traditionally, hardware could only run one computing system at a time, which may not have been a problem for individual users, however, with the growth of digital technology and increased needs for connectivity and data, data centers began adopting virtualization as a solution for providing computing resources efficiently.

An outgrowth of virtualization, container technology is rapidly being adopted by software development organizations. *Linux Containers* allow system designers to bypass traditional operating systems and access computing resources differently. A containerized application relies on a container engine to communicate with the host OS without a hypervisor, or a guest VM. When programmers design these applications, they include all the dependencies needed to perform a specific task in the container. Also, typically, several containers are built which communicate with each other to perform tasks which previously would have been done by one program running on top of an OS.



Doing this provides several advantages, including requiring fewer resources, cutting release cycle times, and abstracting program elements from host operating systems for better portability. Recently, there has been a move to Hybrid Container architecture. This consists of a multi-layered approach where a hypervisor, running on physical hardware, supports virtual machines that, in turn, host containers.

Organizations are weighing the pros and cons of the various configurations available today. Pros include: containers use fewer physical resources than VMs, offer enhanced reliability, and scalability. They also require less storage due to their lightweight nature and offer performance advantages both in startup time and application throughput.

On the con side, when containers are run directly on hardware, they can require considerable effort to ensure segregation and maintain security. When running on shared resources, the container engine needs to be tuned to avoid interference between and within containers. The engine also needs to control access to resources such as memory, I/O devices, and network connections. Finally, when programmers are allowed to create these small, stand-alone resources, it becomes easier for their numbers to get out of control. The advantages of containerization can quickly be outweighed by time and effort spent managing and maintaining them if discipline isn’t exercised from the beginning. Fortunately, the open source community has embraced container development and the tools for designing, deploying, and maintaining them continue to grow at a rapid pace.