What are Containers?

Containers are a form of operating system virtualization. A single container might be used to run anything from a small microservice or software process to a larger application. Inside a container are all the necessary executables, binary code, libraries, and configuration files. Compared to server or machine virtualization approaches, however, containers do not contain operating system images. This makes them more lightweight and portable, with significantly less overhead. In larger application deployments, multiple containers may be deployed as one or more container clusters. Such clusters might be managed by a container orchestrator such as Kubernetes.

Benefits of containers

Containers are a streamlined way to build, test, deploy, and redeploy applications on multiple environments from a developer's local laptop to an on-premises data center and even the cloud. Benefits of containers include:

Less overhead
Containers require fewer system resources than traditional or hardware virtual machine environments because they don't include operating system images.

Increased portability
Applications running in containers can be deployed easily to multiple different operating systems and hardware platforms.

More consistent operation
DevOps teams know applications in containers will run the same, regardless of where they are deployed.

Greater efficiency
Containers allow applications to be more rapidly deployed, patched, or scaled.

Container use cases

Common ways organizations use containers include:

"Lift and shift" existing applications into modern cloud architectures
Some organizations use containers to migrate existing applications into more modern environments. While this practice delivers some of the basic benefits of operating system virtualization, it does not offer the full benefits of a modular, container-based application architecture.

Refactor existing applications for containers
Although refactoring is much more intensive than lift-and-shift migration, it enables the full benefits of a container environment.

Develop new container-native applications
Much like refactoring, this approach unlocks the full benefits of containers.

Provide better support for microservices architectures
Distributed applications and microservices can be more easily isolated, deployed, and scaled using individual container building blocks.

Provide DevOps support for continuous integration and deployment (CI/CD)
Container technology supports streamlined build, test, and deployment from the same container images.

Provide easier deployment of repetitive jobs and tasks
Containers are being deployed to support one or more similar processes, which often run in the background, such as ETL functions or batch jobs.

How do docker and Kubernetes relate to containers?

Users involved in container environments are likely to hear about two popular tools and platforms used to build and manage containers. These are Docker and Kubernetes.

Docker is a popular runtime environment used to create and build software inside containers. It uses Docker images (copy-on-write snapshots) to deploy containerized applications or software in multiple environments, from development to test and production. Docker was built on open standards and functions inside most common operating environments, including Linux, Microsoft Windows, and other on-premises or cloud-based infrastructures.

Containerized applications can get complicated, however. When in production, many might require hundreds to thousands of separate containers in production. This is where container runtime environments such as Docker benefit from the use of other tools to orchestrate or manage all the containers in operation.

One of the most popular tools for this purpose is Kubernetes, a container orchestrator that recognizes multiple container runtime environments, including Docker.

Kubernetes orchestrates the operation of multiple containers in harmony together. It manages areas like the use of underlying infrastructure resources for containerized applications such as the amount of compute, network, and storage resources required. Orchestration tools like Kubernetes make it easier to automate and scale container-based workloads for live production environments.

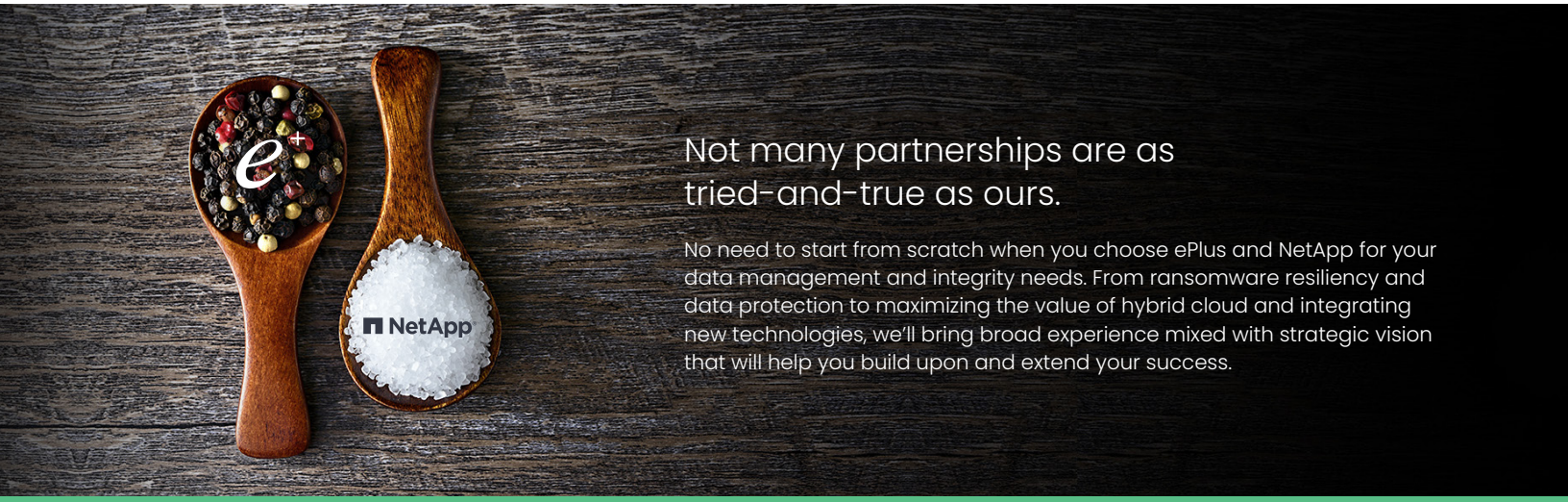
Containers vs. virtual machines (VMs)

People sometimes confuse container technology with virtual machines (VMs) or server virtualization technology. Although there are some basic similarities, containers are very different from VMs.

Virtual machines run in a hypervisor environment where each virtual machine must include its own guest operating system inside it, along with its related binaries, libraries, and application files. This consumes a large amount of system resources and overhead, especially when multiple VMs are running on the same physical server, each with its own guest OS.

In contrast, each container shares the same host OS or system kernel and is much lighter in size, often only megabytes. This often means a container might take just seconds to start (versus the gigabytes and minutes required for a typical VM).

For information on how NetApp is working on proven tools and innovations that deliver and manage persistent storage for any application, in any location, contact [contact info].



Not many partnerships are as tried-and-true as ours.

No need to start from scratch when you choose ePlus and NetApp for your data management and integrity needs. From ransomware resiliency and data protection to maximizing the value of hybrid cloud and integrating new technologies, we'll bring broad experience mixed with strategic vision that will help you build upon and extend your success.