NETWORKWORLD

# What's the future of server virtualization?

## Server virtualization can help fight server sprawl, make better use of compute power, curb energy bills, and improve data-center agility and flexibility.

By Neal Weinberg
Contributor, Network World
JUL 11, 2018 4:12 AM PT

Server virtualization is one of those technologies that's simple in concept and profound in its impact on enterprise data centers.

What if, instead of running one operating system instance and one application per server, you could add a layer of software, known as a hypervisor, that enables you to run multiple operating system instances and associated workloads on a single physical server?

**[ See where SDN is going and learn the difference between SDN and NFV. | Get regularly scheduled insights by signing up for Network World newsletters. ]**

That's the idea behind server virtualization, and the idea dates back to IBM mainframes in the 1960s and was popularized by VMware, which introduced virtualization software for x86 servers in the early 2000s. Since then, other vendors have developed their own server-virtualization platforms and the industry as a whole has created advanced management, automation and orchestration tools that make deploying, moving and managing virtual machine (VM) workloads a breeze.

Prior to server virtualization, enterprises dealt with server sprawl, with underutilized compute power, with soaring energy bills, with manual processes and with general inefficiency and inflexibility in their data-center environments.

Server virtualization changed all that and has been widely adopted. In fact, it's hard to find an enterprise today that isn't already running most of its workloads in a VM environment.

But, as we know, no technology is immune to being knocked off its perch by the next big thing. In the case of server virtualization, the next big thing is going small.

Server virtualization took a physical device and sliced it up, allowing multiple operating systems and multiple full-blown applications to draw on the underlying compute power.

In the next wave of computing, developers are slicing applications into smaller microservices which run in lightweight containers, and also experimenting with serverless computing (also known as function-as-a-service (FaaS).

In both of these scenarios, the VM is bypassed altogether and code runs on bare metal.

## Benefits of server virtualization

The benefits of server virtualization are many, starting with basic server consolidation. You can combine multiple applications on a single piece of hardware, thereby reducing the total number of servers required in the data center. Fewer servers, fewer racks, less networking gear; it all translates into money savings on everything from physical space to maintenance costs to air conditioning.

Server virtualization reduces the need for capital expenditures on new hardware, getting you off that hardware refresh merry-go-round. And you can re-deploy those suddenly freed-up servers.

Remember when data-center admins had to provision servers by hand? With server virtualization comes advances in automation that allow you to spin up a VM in seconds and to move multiple workloads at the touch of a button in response to changing business needs.

Server virtualization also delivers the high availability, failover, speed, scalability, agility, performance and flexibility that today's web-based, highly connected businesses require. And server virtualization is the underlying technology that enables cloud computing vendors to offer

their services. When a customer orders up infrastructure-as-a-service (IaaS) from a cloud service provider, they start with VMs and add on the associated storage, management and security features required to accomplish the task at hand.

## The different types of server virtualization

In the server virtualization world, the physical server is referred to as the host and runs a host operating system. Each VM is a guest and runs a guest operating system. Guests are partitioned from each other.

- With standard hypervisor-based virtualization, the hypervisor or virtual machine monitor (VMM) sits between the host OS and the underlying hardware layer, providing the necessary resources to the guest OSes
- Para virtualization and full virtualization modify the guest operating system before installation into the virtual machine. This enhances performance as the modified guest operating system communicates directly with the hypervisor, eliminating emulation overhead.
- Hardware-assisted virtualization also attempts to reduce hypervisor overhead, but does so through hardware extensions, rather than software modifications.
- With kernel-level virtualization, instead of using a hypervisor, you run a separate version of the Linux kernel. This makes it easy to run multiple virtual machines on a single host, with a device driver used for communication between the main Linux kernel and the virtual machines.
- Finally, with system level or OS virtualization you can run multiple but logically distinct environments on a single instance of the operating system kernel. With system level virtualization, all VMs must share the same copy of the operating system, while server virtualization allows different VMs to have different operating systems.

**[ See also: Will containers kill the virtual machine? ]**

## Virtual machines vs. containers

The two major enablers of the containerization movement are Docker, a popular tool for spinning up containers, and Google's Kubernetes, which helps manage multiple containers. Containers are self-contained code-execution environments that share the kernel of the host OS.

Containers are more streamlined and lightweight than VMs because they bypass the redundant guest OSes and the associated startup overhead. Developers can run as many as six to eight times as many containers as VMs on the same hardware.

Containers do have their downsides. As a relatively new approach, they don't have the wealth of managements tools that a mature technology would have, so there's a lot of set-up and maintenance work that needs to be done. There are also concerns about security.

With VMs, you can easily move workloads between hosts using guest images, but bare metal machines are more difficult to upgrade or move. With bare metal servers, rolling back a machine state is a challenging task.

**[ See also: Serverless explainer: The next generation of cloud infrastructure ]**

# Virtual machines vs. serverless computing

In a traditional IaaS cloud environment, customers first provision VMs, storage, databases and associated security and management tools, then they load applications onto the VMs.

With serverless computing, developers write code and the cloud service provider handles everything else. The developer never has to think about servers, operating systems, provisioning or managing. Of course, there is a physical server that runs the code, but that's the cloud service provider's responsibility.

Instead of a monolithic application, code is broken down into specific functions. When an event happens that triggers that function, the serverless service – for example Amazon's Lambda – runs it. Serverless providers charge customers by the function.

As with the microservice/container scenario, serverless computing bypasses the virtual machine layer and functions run on bare metal. At this point, serverless computing is relatively immature and use cases are limited.

# Future of server virtualization

While containers are hot and interest in serverless computing is growing, the reality is that server virtualization is a rock-solid technology that powers the vast majority of enterprise applications – some estimates put VM saturation as high as 90 percent.

It's difficult to envision an enterprise moving mission critical applications running smoothly on VMs to either containers or a serverless platform. Users with heterogeneous environments will likely still use VMs because containers need to run all on the same OS and can't be mixed between Linux and Windows.

But for new applications that are being built with the latest DevOps and agile methodologies, developers now have options. Going forward, developers will make case-by-case decisions on whether to run new workloads in a traditional VM, a container or a serverless environment.

*Join the Network World communities on Facebook and LinkedIn to comment on topics that are top of mind.*

---

*Neal Weinberg is a freelance technology writer and editor. He can be reached at neal@misterwrite.net.*

*Follow*

**Now read: Getting grounded in IoT**