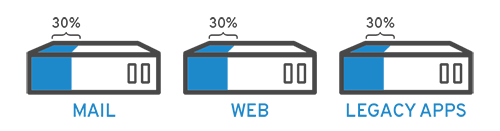
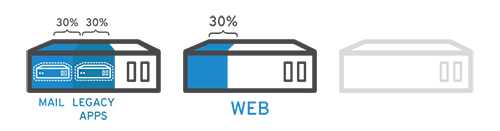
**What is virtualization?**

[Virtualization](https://www.redhat.com/en/topics/virtualization) is technology that lets you create useful IT services using resources that are traditionally bound to hardware. It allows you to use a physical machine’s full capacity by distributing its capabilities among many users or environments.

*In more practical terms*, imagine you have 3 physical servers with individual dedicated purposes. One is a mail server, another is a web server, and the last one runs internal legacy applications. Each server is being used at about 30% capacity—just a fraction of their running potential. But since the legacy apps remain important to your internal operations, you have to keep them and the third server that hosts them, right?



Traditionally, yes. It was often easier and more reliable to run individual tasks on individual servers: 1 server, 1 operating system, 1 task. It wasn’t easy to give 1 server multiple brains. But with virtualization, you can split the mail server into 2 unique ones that can handle independent tasks so the legacy apps can be migrated. It’s the same hardware, you’re just using more of it more efficiently.



Keeping security in mind, you could split the first server again so it could handle another task—increasing its use from 30%, to 60%, to 90%. Once you do that, the now empty servers could be reused for other tasks or retired altogether to reduce cooling and maintenance costs.

**A brief history of virtualization**

While virtualization technology can be [sourced back to the 1960s](https://www.youtube.com/watch?v=u6mu5B-YZU8), it wasn’t widely adopted until the early 2000s. The technologies that enabled virtualization—like [hypervisors](https://en.wikipedia.org/wiki/Hypervisor)—were developed decades ago to give multiple users simultaneous access to computers that performed batch processing. Batch processing was a popular computing style in the business sector that ran routine tasks thousands of times very quickly (like payroll).

But, over the next few decades, other solutions to the many users/single machine problem grew in popularity while virtualization didn’t. One of those other solutions was time-sharing, which isolated users within operating systems—inadvertently leading to other operating systems like [UNIX](http://www.unix.org/what_is_unix.html), which eventually gave way to [Linux®](https://www.redhat.com/en/topics/linux). All the while, virtualization remained a largely unadopted, niche technology.

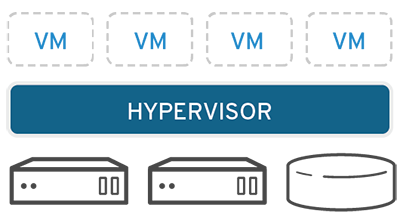
Fast forward to the the 1990s. Most enterprises had physical servers and single-vendor IT stacks, which didn’t allow legacy apps to run on a different vendor’s hardware. As companies updated their IT environments with less-expensive commodity servers, operating systems, and applications from a variety of vendors, they were bound to underused physical hardware—each server could only run 1 vendor-specific task.

This is where virtualization really took off. It was the natural solution to 2 problems: companies could partition their servers *and* run legacy apps on multiple operating system types and versions. Servers started being used more efficiently (or not at all), thereby reducing the costs associated with purchase, set up, cooling, and maintenance.

Virtualization’s widespread applicability helped reduce vendor lock-in and made it the foundation of [cloud computing](https://www.redhat.com/en/topics/cloud). It’s so prevalent across enterprises today that specialized [virtualization management software](https://www.redhat.com/en/topics/virtualization/what-is-virtualization-management) is often needed to help keep track of it all.

**How does virtualization work?**

Software called [hypervisors](https://www.redhat.com/en/topics/virtualization/what-is-a-hypervisor) separate the physical resources from the virtual environments—the things that need those resources. Hypervisors can sit on top of an operating system (like on a laptop) or be installed directly onto hardware (like a server), which is how most enterprises virtualize. Hypervisors take your physical resources and divide them up so that virtual environments can use them.

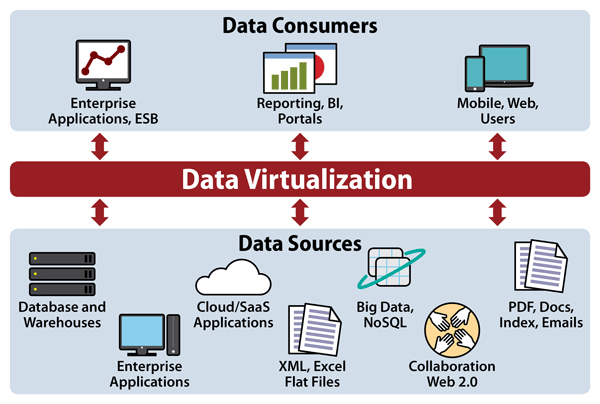


Resources are partitioned as needed from the physical environment to the many virtual environments. Users interact with and run computations within the virtual environment (typically called a guest machine or [virtual machine](https://www.redhat.com/en/topics/virtualization/what-is-a-virtual-machine)). The virtual machine functions as a single data file. And like any digital file, it can be moved from one computer to another, opened in either one, and be expected to work the same.

When the virtual environment is running and a user or program issues an instruction that requires additional resources from the physical environment, the hypervisor relays the request to the physical system and caches the changes—which all happens at close to native speed (particularly if the request is sent through an open source hypervisor based on KVM, the [Kernel-based Virtual Machine](https://www.redhat.com/en/topics/virtualization/what-is-KVM)).

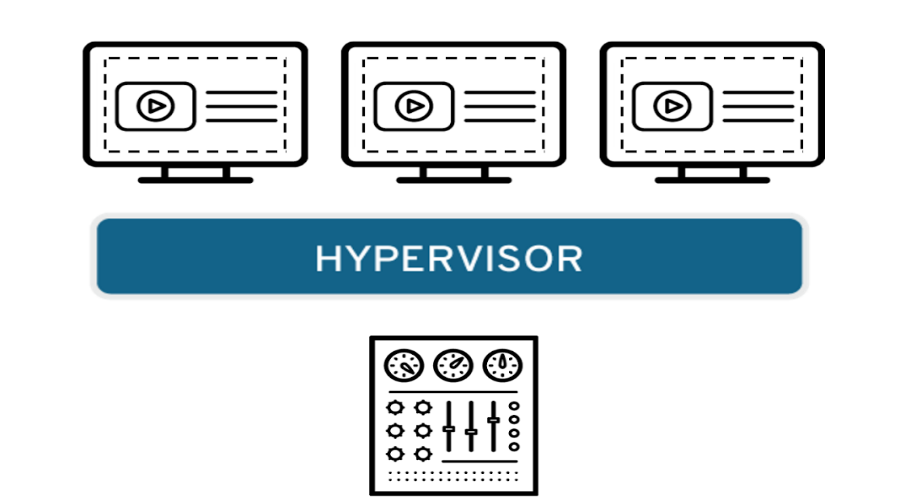
**Types of virtualization**

**Data virtualization**



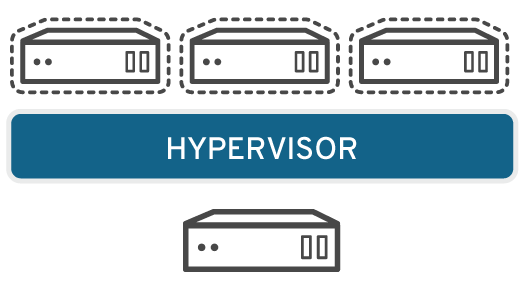
Data that’s spread all over can be consolidated into a single source. Data virtualization allows companies to treat data as a dynamic supply—providing processing capabilities that can bring together data from multiple sources, easily accommodate new data sources, and transform data according to user needs. Data virtualization tools sit in front of multiple data sources and allows them to be treated as single source, delivering the needed data—in the required form—at the right time to any application or user.

**Desktop virtualization**



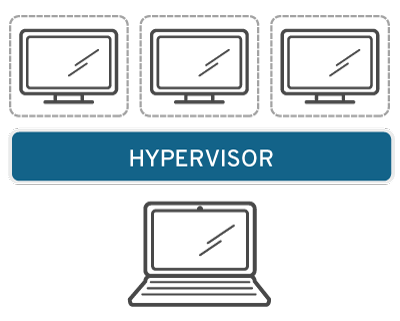
Easily confused with operating system virtualization—which allows you to deploy multiple operating systems on a single machine—desktop virtualization allows a central administrator (or automated administration tool) to deploy simulated desktop environments to hundreds of physical machines at once. Unlike traditional desktop environments that are physically installed, configured, and updated on each machine, desktop virtualization allows admins to perform mass configurations, updates, and security checks on all virtual desktops.

**Server virtualization**



Servers are computers designed to process a high volume of specific tasks really well so other computers—like laptops and desktops—can do a variety of other tasks. Virtualizing a server lets it to do more of those specific functions and involves partitioning it so that the components can be used to serve multiple functions.

**Operating system virtualization**

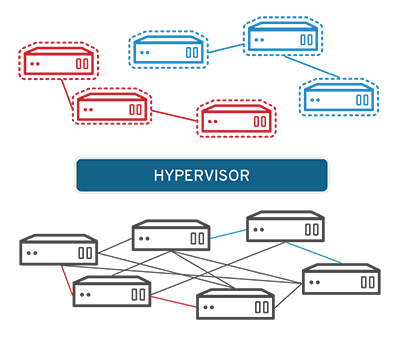


Operating system virtualization happens at the [kernel](https://www.linux.com/blog/what-linux-kernel)—the central task managers of operating systems. It’s a useful way to run Linux and Windows environments side-by-side. Enterprises can also push virtual operating systems to computers, which:

* Reduces bulk hardware costs, since the computers don’t require such high out-of-the-box capabilities.
* Increases security, since all virtual instances can be monitored and isolated.
* Limits time spent on IT services like software updates.

[Learn more about operating system virtualization](https://en.wikipedia.org/wiki/Operating-system-level_virtualization)

**Network functions virtualization**



Network functions virtualization (NFV) separates a network's key functions (like directory services, file sharing, and IP configuration) so they can be distributed among environments. Once software functions are independent of the physical machines they once lived on, specific functions can be packaged together into a new network and assigned to an environment. Virtualizing networks reduces the number of physical components—like switches, routers, servers, cables, and hubs—that are needed to create multiple, independent networks, and it’s particularly popular in the telecommunications industry.