SEI Insights

Home > **SEI Blog** > Multicore and Virtualization: An Introduction

SEI Blog

The Latest Research in Software Engineering and Cybersecurity

■ Multicore and Virtualization: An Introduction

POSTED ON AUGUST 14, 2017 BY **DONALD FIRESMITH** [/AUTHOR/DONALD-FIRESMITH] IN **MULTICORE PROCESSING AND VIRTUALIZATION** [HTTPS://INSIGHTS.SEI.CMU.EDU/SEI_BLOG/MULTICORE-PROCESSING-AND-VIRTUALIZATION/]

By Donald Firesmith
Principal Engineer
Software Solutions Division

Multicore processing and virtualization are rapidly becoming ubiquitous in software development. They are widely used in the commercial world, especially in large data centers supporting cloud-based computing, to (1) isolate application software from hardware and operating systems, (2) decrease hardware costs by enabling different applications to share underutilized computers or processors, (3) improve reliability and robustness by limiting fault and failure propagation and support failover and recovery, and (4) enhance scalability and responsiveness through the use of actual and virtual concurrency in architectures, designs, and implementation languages. Combinations of multicore processing and virtualization are also increasingly being used to build mission-critical, cyber-physical systems to achieve these benefits and leverage new technologies, both during initial development and technology refresh.

In this introductory blog post, I lay the foundation for the rest of the series by defining the basic concepts underlying multicore processing and the two main types of virtualization: (1) virtualization by virtual machines and hypervisors and (2) virtualization by containers. I will then briefly compare the three technologies and end by listing some key technical challenges these technologies bring to system and software development.

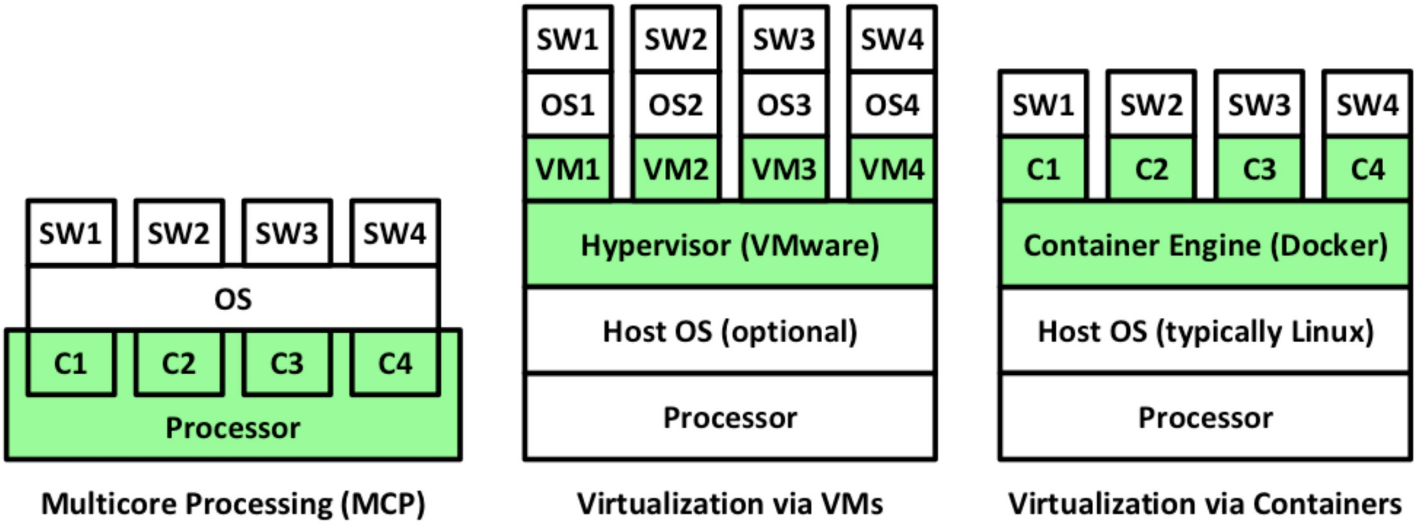
Underlying Concepts

Multicore processing (MCP) is computer processing performed using multicore processors. A **multicore processor** [https://en.wikipedia.org/wiki/Multi-core_processor] is a single integrated circuit (a.k.a., chip multiprocessor or CMP) that enables software applications to run on multiple real core processing units, more commonly known as *cores*.

Virtualization [https://en.wikipedia.org/wiki/Virtualization] is a collection of software technologies that enable software applications to run on *virtual hardware* (virtualization via virtual machines and hypervisor) or *virtual operating systems* (virtualization via containers). A **virtual machine (VM)** [https://en.wikipedia.org/wiki/Virtual_machine], also called a guest machine, is a software simulation of a hardware platform that provides a virtual operating environment for guest operating systems. A **hypervisor** [https://en.wikipedia.org/wiki/Hypervisor], also called a virtual machine monitor (VMM), is a software program that runs on an actual host hardware platform and supervises the execution of the guest operating systems on the virtual machines. A **container** [https://en.wikipedia.org/wiki/Operating-system-level_virtualization] is a virtual runtime environment that runs atop a single OS kernel without emulating the underlying hardware.

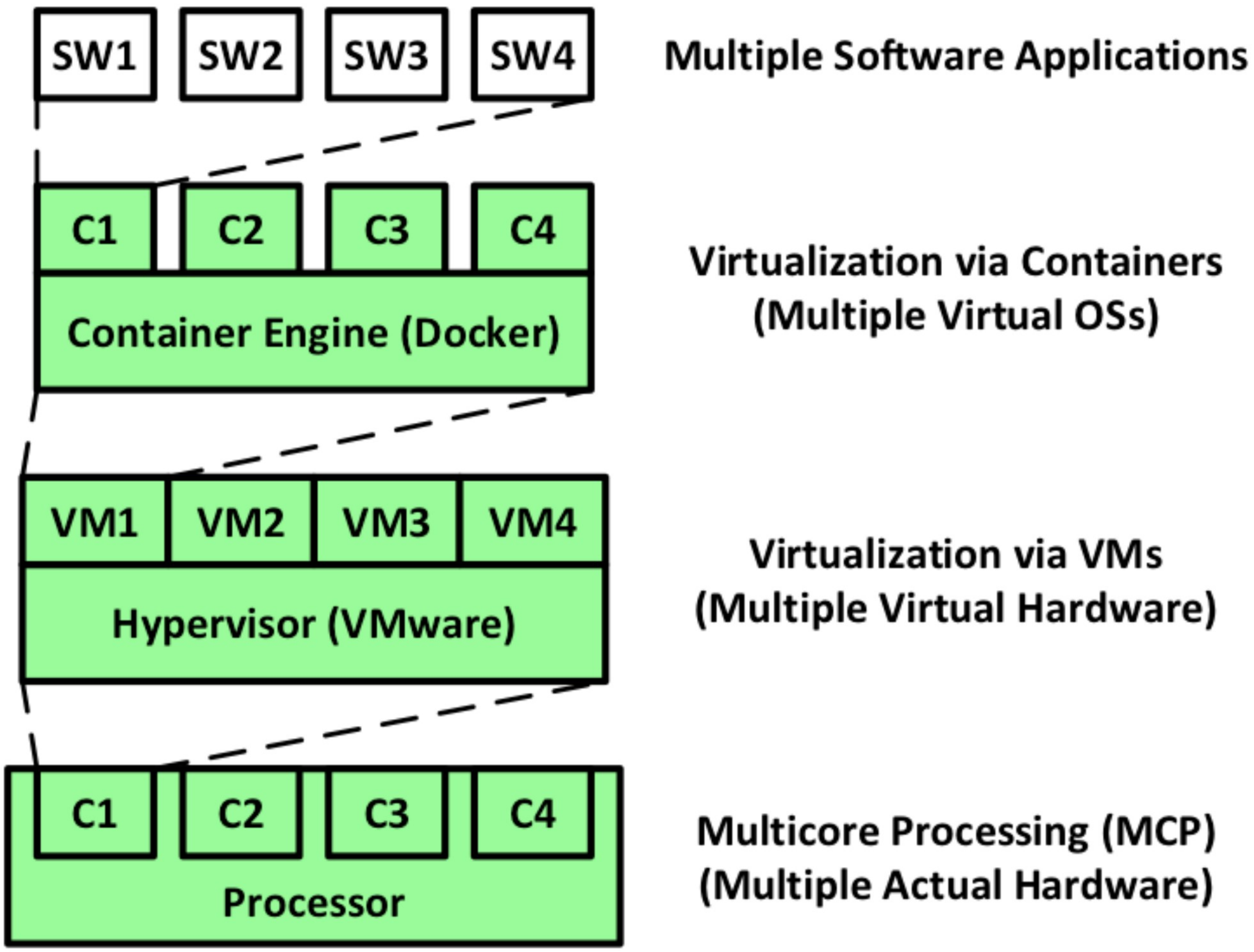
Comparison

The following figure shows the architectural differences between multicore processing, virtualization via VM, and virtualization via container, with the relevant technologies highlighted in green. In this figure, SW1 through SW4 represent four different software applications. The standard acronyms OS and VM stand for operating system and virtual machine, respectively. The boxes labeled C1 through C4 on the left represent cores in a multi-core processor, whereas on the right they represent containers. The figure also implies the multi-core processing is primarily a hardware technology, whereas virtualization, whether by virtual machines or containers, is a software technology.



The following figure provides another way of looking at the three technologies: by considering the layer at which the technology exists. Multicore processing provides *multiple real hardware platforms*. Virtualization by VMs sits on top of multicore processing and provides *multiple virtual hardware platforms*. Virtualization by containers can sit on top of virtual machines where it can provide *multiple virtual operating systems*. This figure also shows that:

- Multiple software applications can be allocated to a single container.
- Multiple containers can be allocated to a single virtual machine.
- Multiple virtual machines can be allocated to a single core.
- Multiple cores are contained by a single multicore processor.


SW1
SW2
SW3
SW4
Multiple Software Applications
C1
C2
C3
C4
Container Engine (Docker)
Virtualization via Containers
(Multiple Virtual OSs)
VM1
VM2
VM3
VM4
Hypervisor (VMware)
Virtualization via VMs
(Multiple Virtual Hardware)
C1
C2
C3
C4
Processor
Multicore Processing (MCP)
(Multiple Actual Hardware)

Challenges

Although these three technologies offer many important benefits, they also have significant (potentially) negative ramifications that system and software architects must address:

- Multicore and virtualization add additional complexity to the architecture, which affects both analysis (e.g., performance, safety, and security) and testing. Traditional analysis may yield false results, and proper analysis becomes more complex. Testing must uncover different bugs and localization of defects may become harder.
- These technologies add layers of shared resources (e.g., caches, memory controllers, I/O controllers, and busses), which are potential single points of failure and sources of interference between applications.
- The overhead of virtualization, especially virtualization by virtual machines and their associated hypervisor, can significantly increase.
- Systems become less deterministic due to increases in actual and virtual concurrency.

- Current safety and security standards, policies, and architectural patterns are based on assumptions that no longer hold true and thus may be inconsistent with these technologies. These inconsistencies may require changes to accreditation and certification policies, including new ways to verify real-time and/or safety-critical systems.

Future Blog Entries

The next post in this series will define multicore processing, list its current trends, document its pros and cons, and briefly address its safety and security ramifications. The following two blog entries will do the same for virtualization via virtual machines and virtualization via containers. These postings will be followed by a final blog entry providing general recommendations regarding the use of these technologies on mission-, safety-, and security-critical, cyber-physical systems.

Additional Resources

Read **all blog posts by Don Firesmith** [http://insights.sei.cmu.edu/author/donald-firesmith/] .

About the Author

Donald Firesmith

✉ **Contact Donald Firesmith** [https://www.sei.cmu.edu/contact.cfm]
Visit the SEI Digital Library for **other publications by Donald** [http://resources.sei.cmu.edu/library/author.cfm?authorID=4637]
View **other blog posts by Donald Firesmith** [/author/donald-firesmith]

Terms of Use | **Privacy Statement** | **Intellectual Property**

The Software Engineering Institute (SEI) is a federally funded research and development center (FFRDC) sponsored by the U.S. Department of Defense (DoD). It is operated by Carnegie Mellon University.