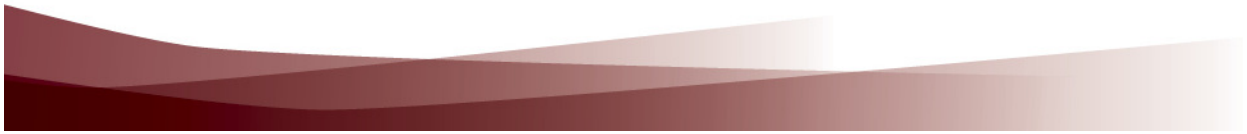
Appendix -
Beyond Hypervisors: Containers and Unikernels

Roch Glitho, PhD

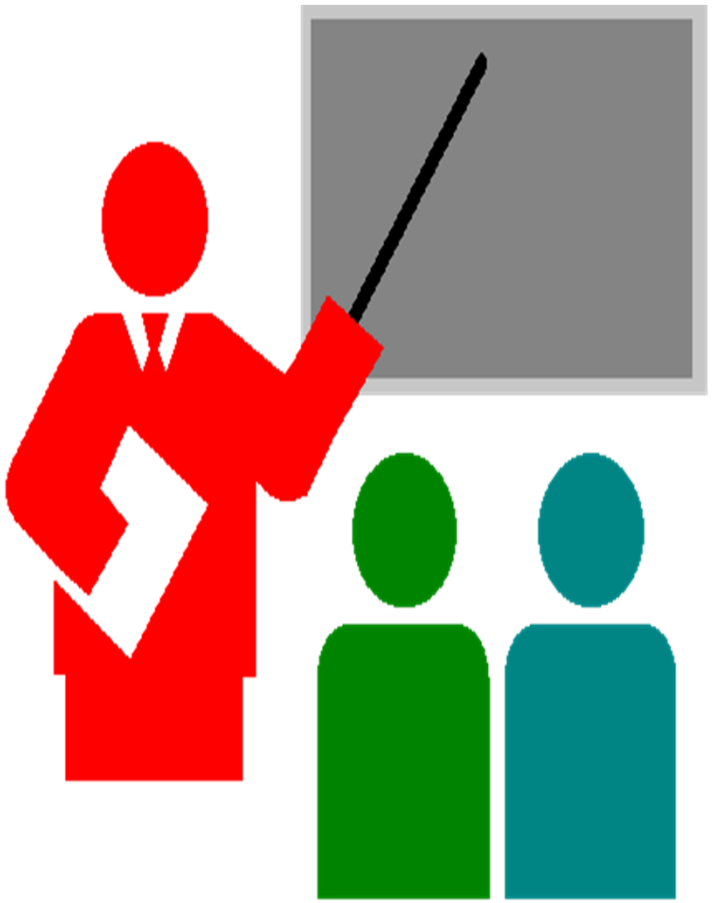
Professor and Canada Research Chair

My URL - http://users.encs.concordia.ca/~glitho/



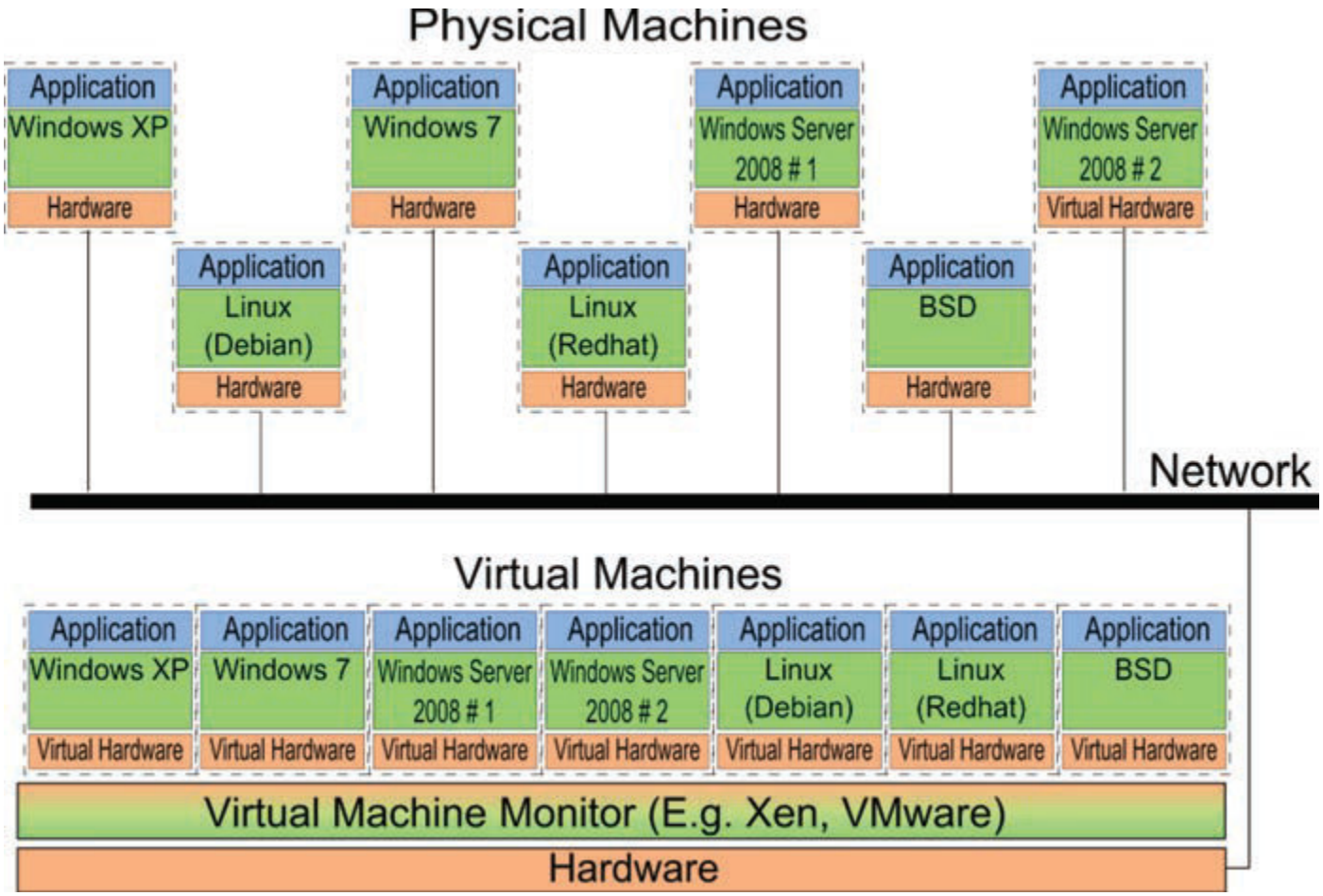
Concordia University
Engineering and Computer Science
Concordia Institute for Information Systems Engineering

Containers and Unikernels



- Issues with hypervisors
- Alternatives (Containers and unikernels)

virtual machines, hypervisors



From reference [1] – Note: There is a small error in the figure

Hypervisor

In a hypervisor based – approach, a VM includes the application + full blown operating system (e.g. Linux Debian, Linux Red Hat)

- OS on virtual machine needs to boot
  - Slow starting time for application
- Resources are not used in an efficient manner
  - Linux kernel replicated in each VM that runs linux.

Proposed Solutions

Back to operating systems basics

- The two components of an operating system
  - Kernel
    - Interacts with the hardware and manages it (e.g. write/read a disk partition)
  - Librairies
    - Set of higher level functions accessible to programs via system calls
      - Enable function like create / read / delete file while hiding the low level operations on the hard disk

Alternatives

VM vs container vs Unikernel

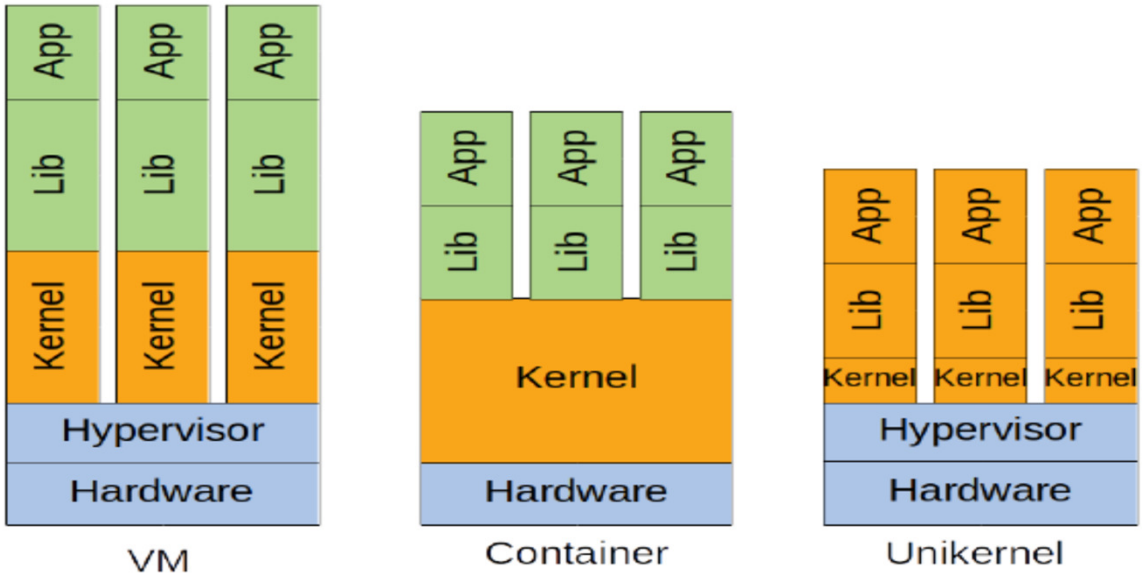


Fig. 1. Comparison of virtual machine, container and unikernel system architecture

T. Goethals et al., Unikernels vs. Containers: An In-Depth Benchmarking Study in the Context of Microservice Application, IEEE SC2 Conference, November 2018

On containers

Operating system (Kernel) virtualization:

- Kernel offers isolated spaces to run containers
  - Containers
    - » Applications packaged with their run time environment that run on a same kernel
      - » Run as processes, but with isolated file system, networking, CPU and memory resources

On containers

Operating system (Kernel) virtualization:

- Kernel offers isolated spaces to run containers
  - Containers
    - » Hosted by container engine (e.g. Docker Engine)
    - » Need to be deployed, managed and orchestrated (e.g. Kubernetes)

On containers

Operating system (Kernel) virtualization:

- **Kernel offers isolated spaces to run containers**
  - **Some pros / cons**
    - Less memory footprint
      - » Do not include kernel
    - Faster start up time
      - » Kernel does not need to boot

On containers

Operating system (Kernel) virtualization:

- **Kernel offers isolated spaces to run containers**
  - **Some pros / cons**
    - Works only in environments in which you have given operating system kernel + its libraries (e.g. Linux kernel + Linux distributions)
    - Less secure than VM
      - » Challenge:
        - » Trade-off between isolation and performance / efficiency

On Unikernels

Application + Tiny run time:

- **Tiny run time**
  - Not the whole OS like VM
  - Not the whole libraries like containers
    - » Only the function required by the applications
    - » Static binding
  - Can run as a tiny VM or a tiny container

On Unikernels

Pros and cons:

- Smaller footprint
- Boot up faster
- Less flexible
  - Addition / removal of functionality requires re-compilation

References

1. . T. Goethals et al., Unikernels vs. Containers: An In-Depth Benchmarking Study in the Context of Microservice Application, IEEE SC2 Conference, November 2018

2. P. Aditya et al, Will Servless Computing Revolutionize NFV, Proceedings of the IEEE, April 2019

The End