I/O Device Virtualization in the x86 Architecture

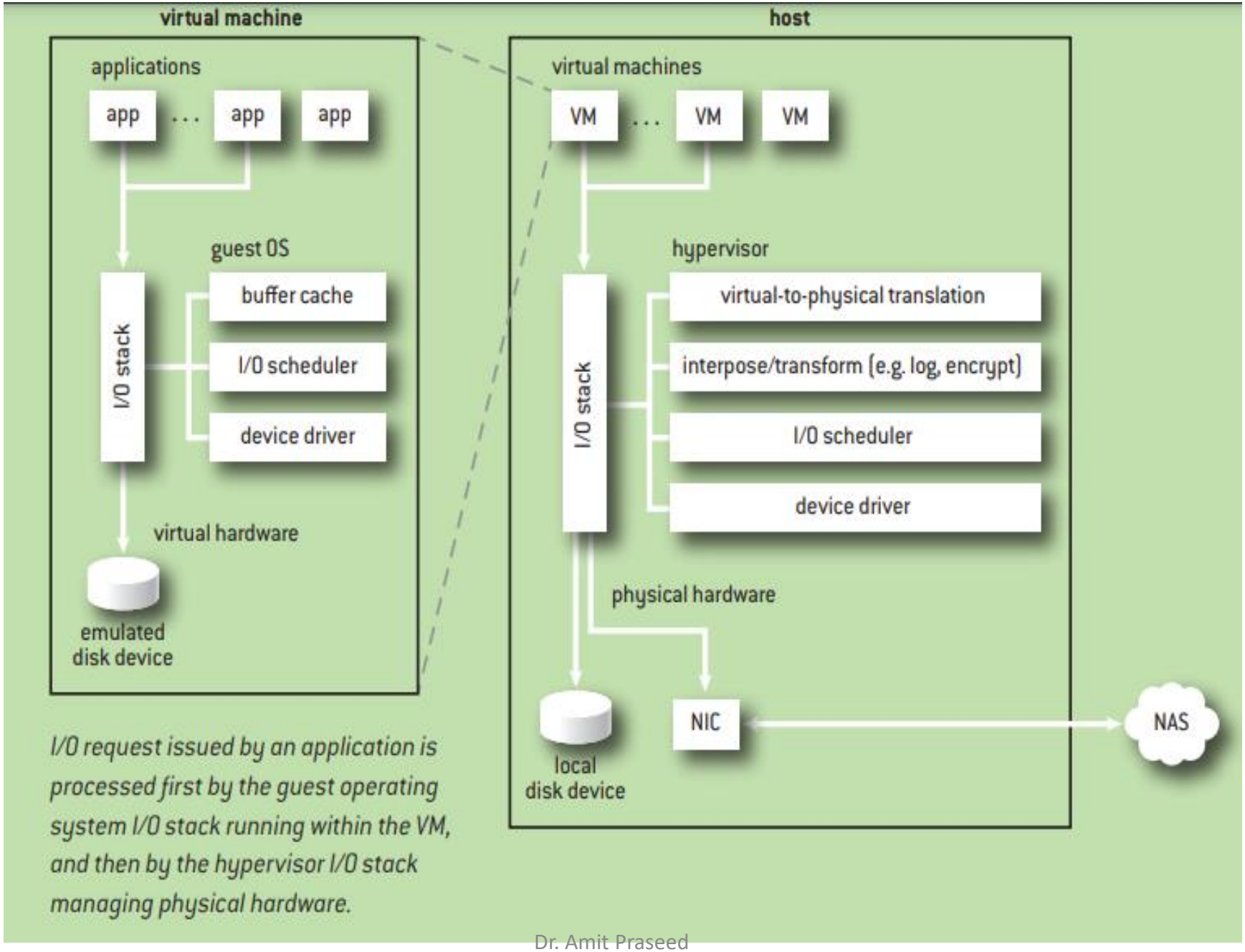
Dr. Amit Praseed

I/O Device Virtualization

- Allows multiple logical devices to be implemented by a smaller number of physical devices
- Provides for flexible mappings between logical and physical devices, facilitating seamless portability
- Device aggregation
  - Multiple physical devices can be combined into a single more capable logical device
- New features can be added to existing systems by interposing and transforming virtual I/O requests, transparently enhancing unmodified software with new capabilities
- Enhanced system security

Challenges

- Defining appropriate semantics for virtual devices and interfaces, especially when faced with complex physical I/O devices or system-level optimizations
- Device contention between multiple VMs
- Achieving good I/O performance despite the potential overhead



I/O request issued by an application is processed first by the guest operating system I/O stack running within the VM, and then by the hypervisor I/O stack managing physical hardware.

Split Driver Architecture

- Most hypervisors employ a split driver architecture for dealing with I/O devices
  - different virtual device interface emulation front-ends
  - multiple different back-end implementations of the device
- The virtualization layer and the virtual-device emulation code must support the architectural and device semantics

Need for Optimization

- Reducing I/O-virtualization overhead requires decreasing virtual-device emulation costs
  - reduce the number of trap-and-emulate operations
  - using virtual hardware optimized for the virtualization layer - paravirtualization
  - Pass through mode – when a single VM uses a device