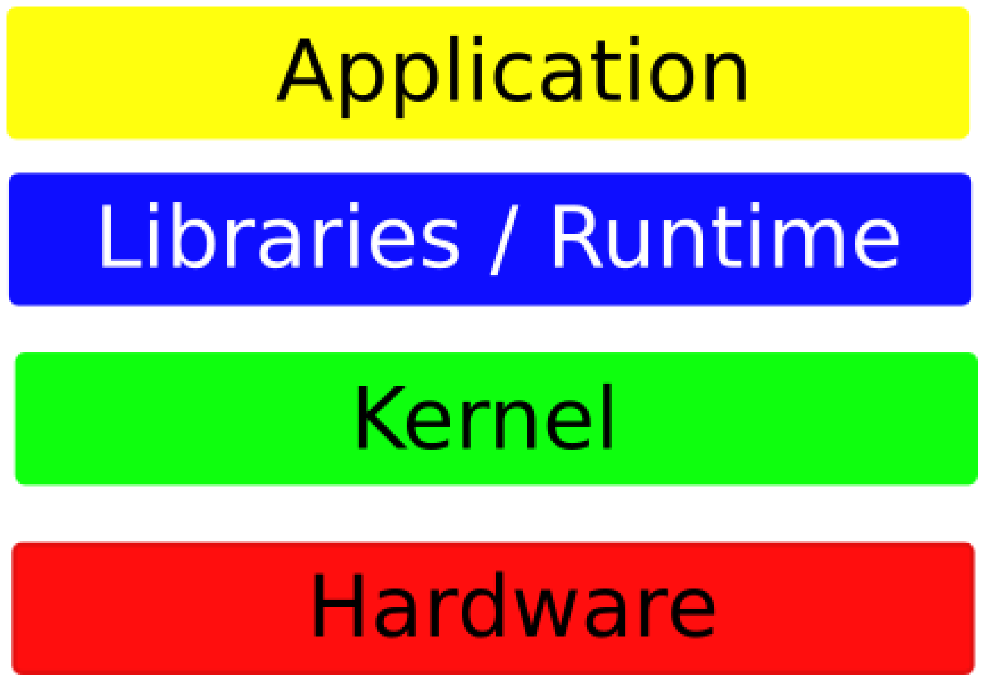
Virtualization and Containers

FOSSASIA 2016

Amit Shah | Red Hat | amit.shah@redhat.com

The Problem

One Server, One Application



Datacenter

- Servers keep getting more and more capable
  - Many cores
  - Multi-gig RAM
- ... but applications don't always use it all
  - Monolithic apps
  - Many 32-bit
  - Don't scale to multiple cores

Datacenter

- Under-used servers means less power efficiency
- New servers need new Operating Systems to utilise features (or even get them to run)
  - Applications may need to be ported to new OS
  - "If it ain't broke, don't fix it"
- Security concerns from multiple applications
  - Exploit in one could lead to compromise of others

Service Providers

- Companies selling computing resources
  - e.g. one can buy VPS servers, webspace
- Need isolation between various partitions
  - Else, one rogue application can bring down other customers' applications on the same host
- Resource Management
  - Partitions should only get resources customers are paying for
    - e.g. 512MB RAM, 2 CPU cores at 2GHz, 5GB storage
- Maximize use of their hardware – increased density

What Admins Want

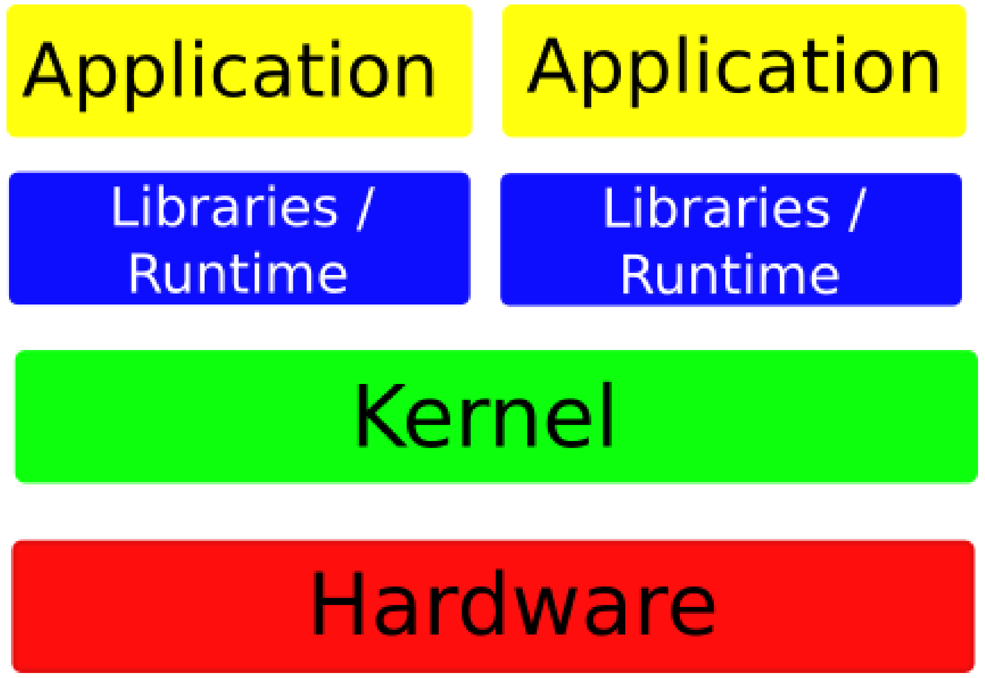
- Isolation for each application
- Clean, pristine environments for applications
- Utilize each server to its full capacity
  - Helps save on power bills and cooling costs
- Maximize use of computing resources
  - CPU, RAM, network
- Applications to continue working without changes
  - Operating Systems to continue working without changes
- Each of these can be achieved with varying levels of complexity and costs
- Deployments made based on risk and cost analysis

Solutions

chroot

- Filesystem-level abstraction
- Useful for
  - 'clean environments'
  - System recovery
  - Package builds
    - verify package scripts mention all build and runtime dependencies
- Not useful for anything else
  - Unsuitable for the datacenter and service provider usecases mentioned earlier

One Server, Multiple Userspaces



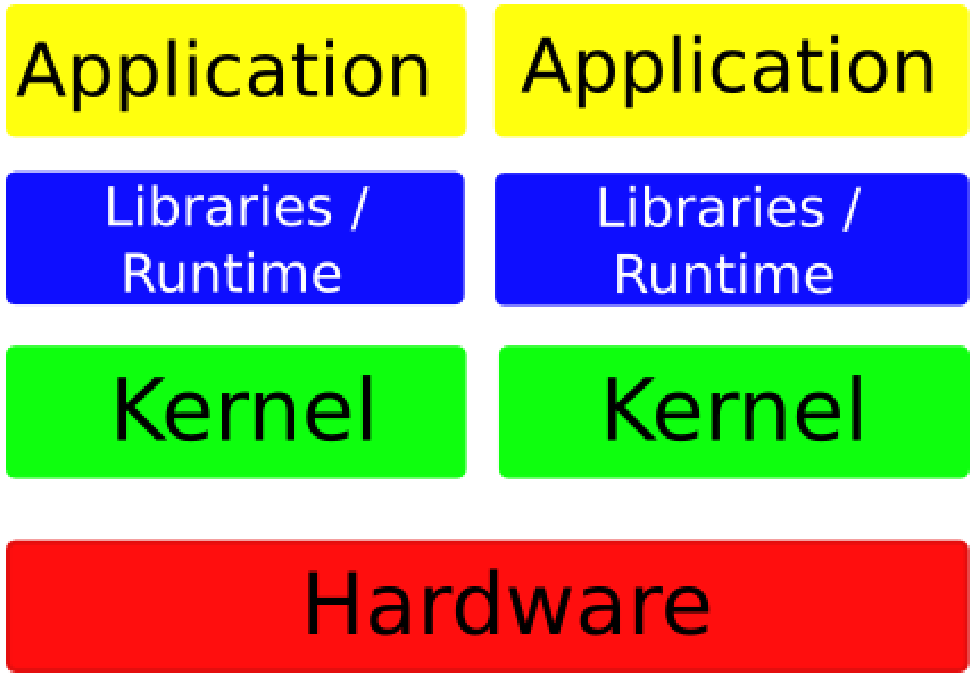
Containers

- Operating System virtualization
- Namespace restrictions
  - Filesystem, PID, process, network, disk
- Resource limits
  - via cgroups

Containers

- Pros
  - Fast load times
  - Very high density / low overhead
  - Fairly reasonable level of isolation*
- Cons
  - Share kernel
    - exploit from one container means others are compromised too
  - Apps and runtime have to run on given kernel

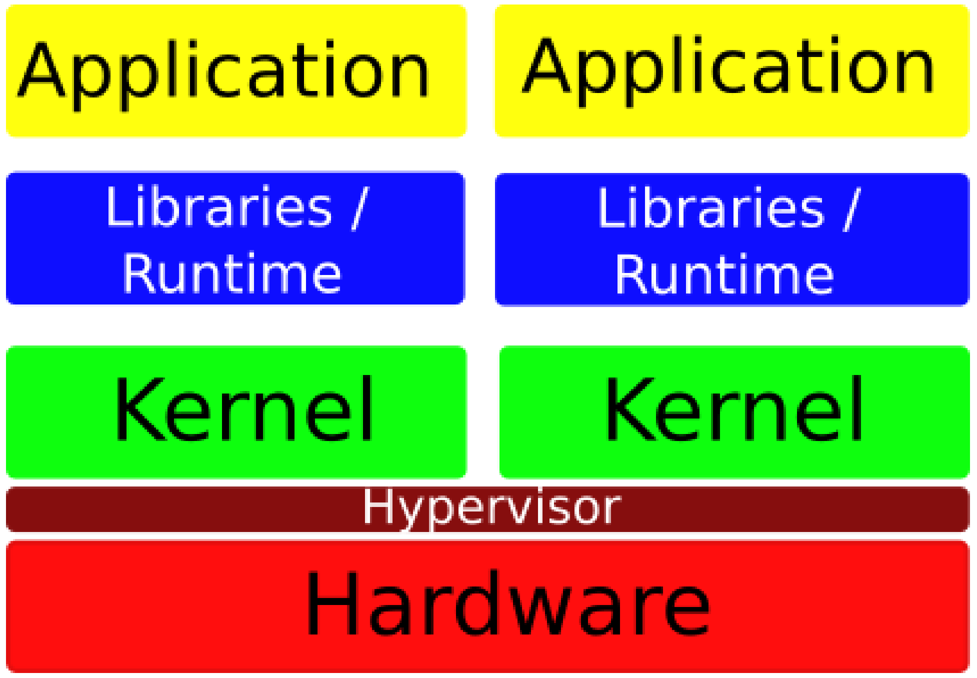
Virtualize the Hardware



Add a Layer!

- Every computing problem can be solved by adding another layer of abstraction

Virtualization



Hardware Virtualization

- Not a new idea!
- Recent (~9 years) advancements in x86 ISA have made this very accessible
  - Enabled open source virtualization solutions
- Existing OSes, apps can be moved to a virtualized environment
  - No changes to the OS or the app
- Each generation of processors making this more performant and more secure
- Enables features like live migration

Hardware Virtualization

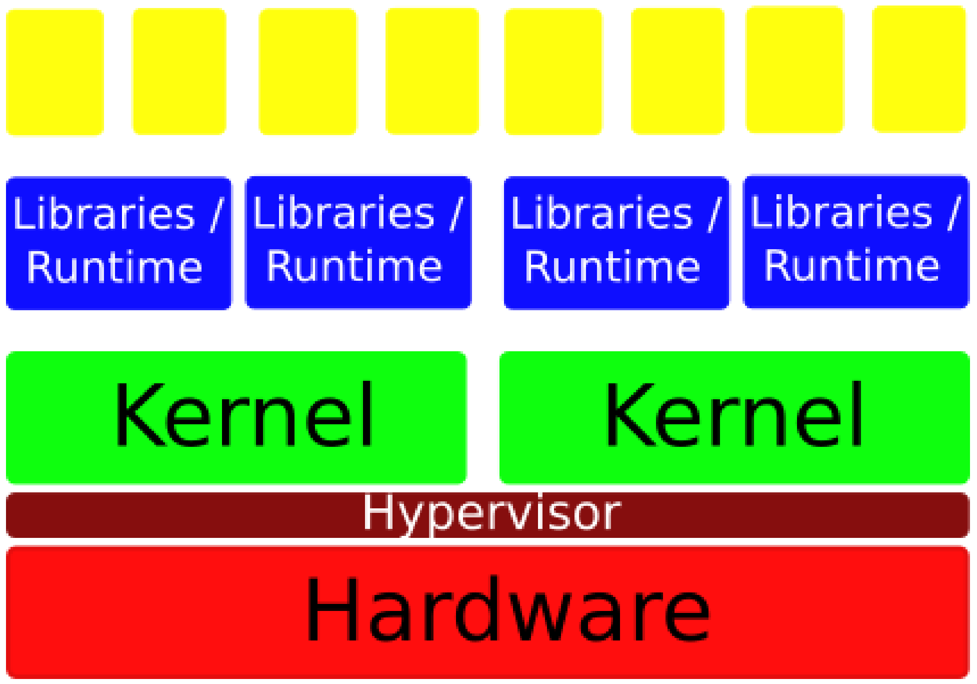
- Pros
  - Most secure
    - Very low shared footprint: hypervisor
    - Kernel / root exploits within guests stay there
  - Overheads keep going lower with h/w advances
- Cons
  - Needs emulation of devices
    - can be slower
  - Lower density than Containers

Hardware Virtualization

- Enabled cloud providers to come up
- Cloud providers themselves used hardware virtualization, and customers could then use OS virtualization (containers) to increase utilization of compute resources

Combining Solutions

Hardware and OS Virtualization



Automation

- Dev side of things
- 'But it runs in my setup!'
- Management of VMs, containers, is a big space, with lots happening in the area
- Docker
  - Kind of appropriated the 'containers' term
  - Really a way to package and manage apps along with runtimes

Bonus

Unikernels

- Stripped-down binaries that run directly on hypervisors / OSes
  - App, runtime all bundled together
- Special-purpose OSes
- Fast boot-up, low footprint

Thank You!

Amit Shah | http://log.amitshah.net | amit.shah@redhat.com