Running Jenkins in Docker

UNDERSTANDING DOCKER AND JENKINS

Chris B. Behrens
SOFTWARE ARCHITECT

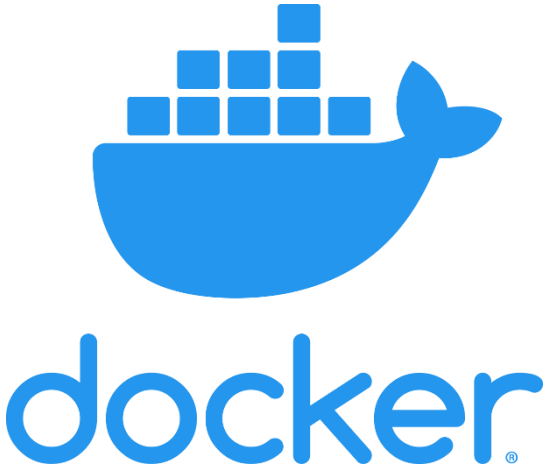
@chrisbbehrens

My Assumptions About You

A strong grasp of Jenkins

Less of a grasp of Docker

Docker and the Kernel



What’s the fundamental difference between a virtual machine and a container?

Kernel

What exactly do you mean by kernel?

Demo

Proving that we're running Windows 10

Start a Jenkins / Linux container

Open a command line to that

Prove that that has a Debian shell

Kernel

A privileged layer of abstraction that exists between applications and hardware.

What Is a Kernel?

"Oh, it's the HAL."

"So, it's the...BIOS?"

"Where do device drivers fall in this?"

Types of Kernels

Linux – monokernel, one kernel for all processes

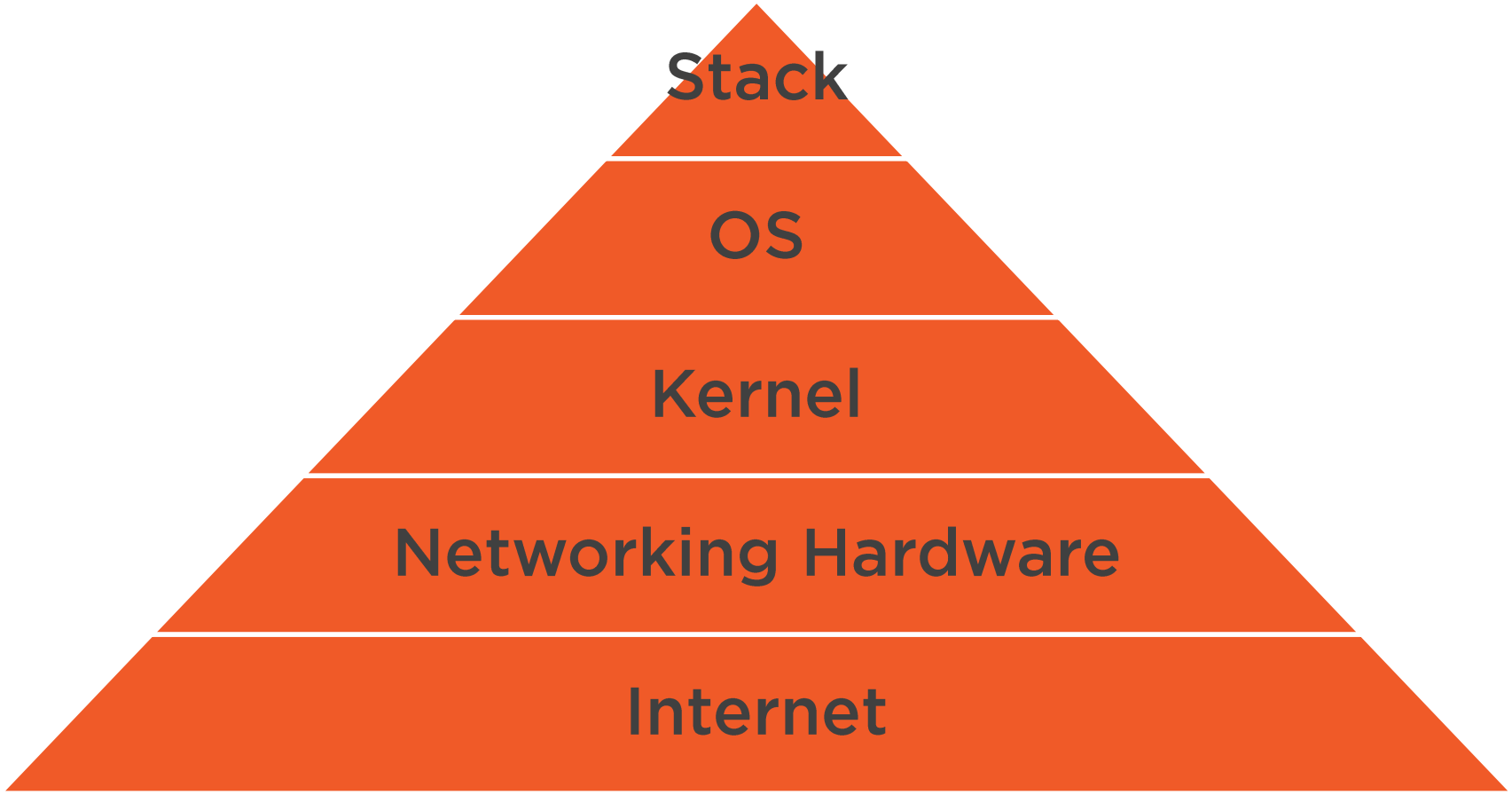
Minix – microkernel, kernel per user space

Windows

Windows – a hybrid of both approaches

If you have kernel level access, you’ve got access to everything.

Kernel-level Access



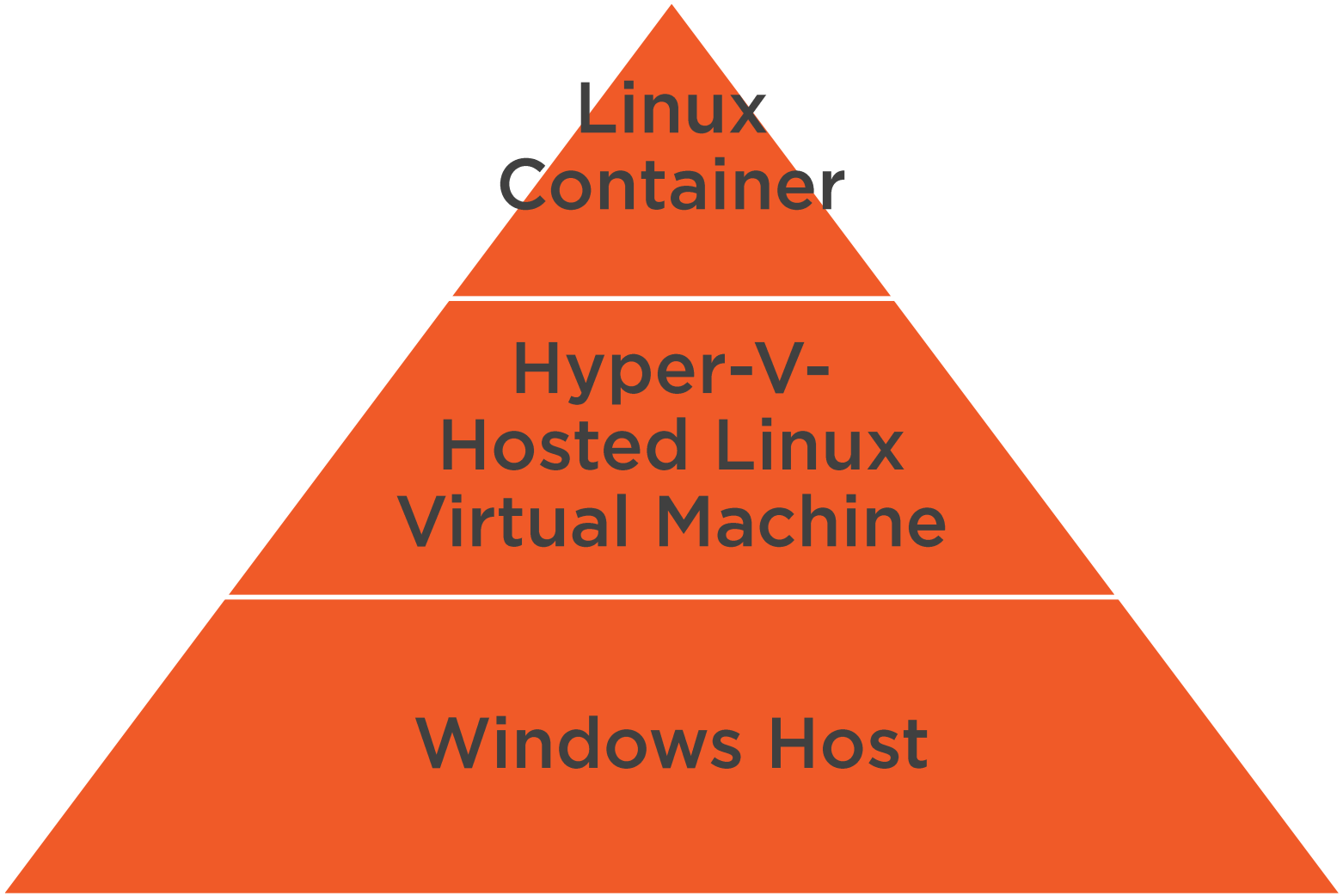
Why This Matters: Running Linux Containers on Windows

Linux Containers on Windows

Linux shell on top of NT Kernel

Container on top of VM on top of Windows*

Linux Containers on Windows: How



Container Considerations

Performance - “containers result in equal or better performance than VMs in almost all cases”

Security – a function of the vulnerabilities of the OS

Portability – no worries

What's the Point of This?

"I'm running Linux on Linux, like a sane person"

"I'm running Jenkins on Windows"

Now you know where your builds, Jenkins, your containers and the kernel all begin and end

Linux Containers on Windows: A Way Forward

Windows Subsystem for Linux 2 (WSL2)

WSL2 - a legit Linux kernel

This is still experimental at the time of recording

Get the VM approach working first, then try out WSL2

The Vision and the Why: Jenkins on Docker

Builds in Azure DevOps

1. Queue a build

2. Azure reads the build definition

- Compiles *demands*
  - OS
  - Dependencies
  - Other

3. Matches the build with a build agent that meets demands

4. Spins up a VM

Subsequent builds are independent of this first process (idempotent)

Queued Builds vs. Parallelized

Universal
Jenkins Agent

```
1.pr#3143-ui-update
2.pr#3145-lclz-crctn
3.qa-integration
```

Queued Builds vs. Parallelized

Specialized Jenkins Agent	Specialized Jenkins Agent	Specialized Jenkins Agent
1. pr#3143-ui-update	1. pr#3145-lclz-crctn	1. qa-integration

Jenkins Queue Models

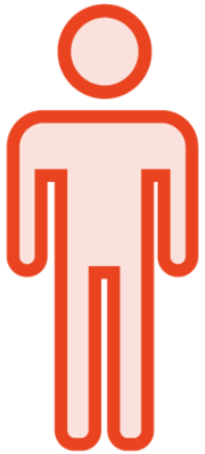
Master-Slave

Dedicated machines

Dedicated VMs

Containers are better on every measure

The Ideal Model



The Docker Plug-in

Jenkins

Docker 1.1.9

Minimum Jenkins requirement: 2.60.3
ID: docker-plugin

Installs: 23224
GitHub →
Last released:4 months ago

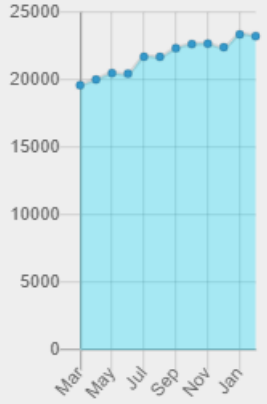
Maintainers

Nigel Magnay
Nicolas De Loof
Peter Darton

Dependencies

Apache HttpComponents Client 4.x API ≥ 4.5.3-2.0
bouncycastle API ≥ 2.16.2
Docker Commons ≥ 1.9
Docker API ≥ 3.0.14
Durable Task ≥ 1.16
SSH Build Agents ≥ 1.22
Token Macro ≥ 2.3
Pipeline: API ≥ 2.23.1 (optional)
Pipeline: Groovy ≥ 2.41 (optional)

Archives
Get past versions



Demo

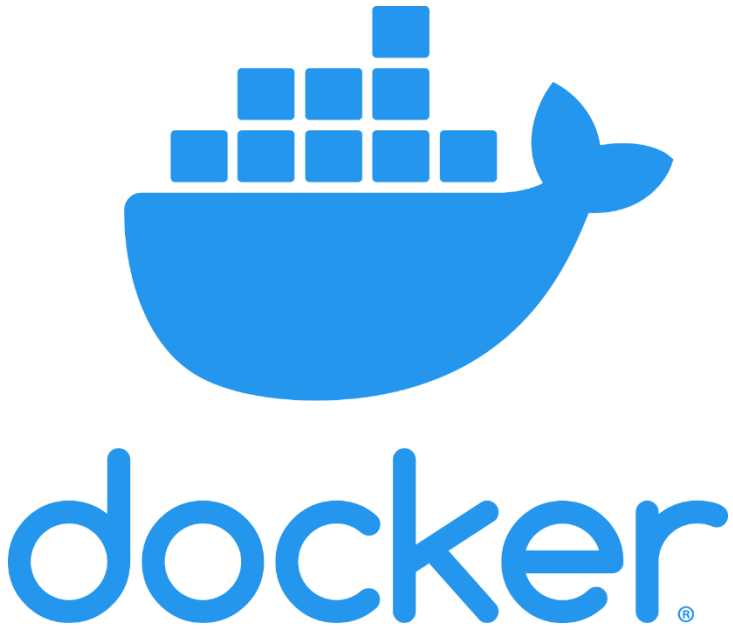
Pull a Jenkins LTS image from DockerHub

Get it running quickly

Modify our docker run command

Tweak some Jenkins options

Maintaining State Outside the Container



Isolation of state from the container

Containers are layers

- OS
- Application
- Configuration
- Which are **read-only**

The topmost layer is writable

Our container will place a writable layer on top of Jenkins:LTS

Layers in Our Container

Take control of the top layer

- Isolated
- Monitored
- Backed up

The most essential form of isolating state: pipeline scripts

Migrate from classic build definitions

To keeping your pipeline scripts in version control

Keep your builds in version control with pipeline scripts

The Docker File System

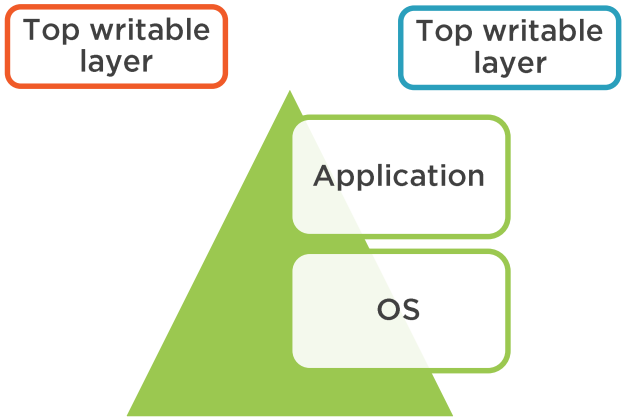
Two containers with the same base image

Container A has default plugins, cbehrens is admin

Container B has custom plugins, jsmith is admin

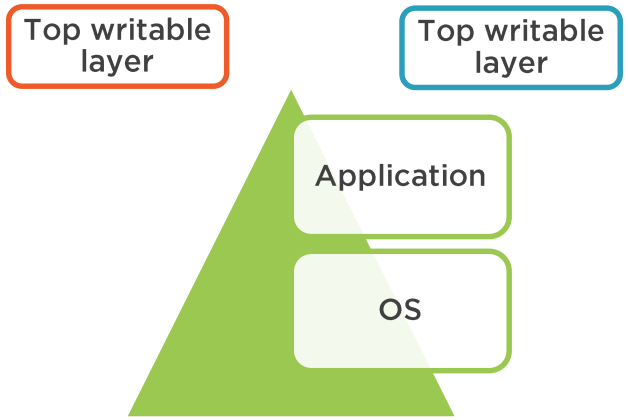
Shared descendant layers, different top-level layers

Two Ways to Represent This



Container – Image = Top Writable Layer

Understanding Copy on Write



1. Two full copies of the file system

- Or maybe a writeable subset
- Stored independently

2. A single copy of the file system

- Only the deltas are stored
- When the file changes
  - We find it
  - Copy it to the TWL
  - Persist the changes

A Jenkins example

- Modify config file
- Storage driver searches the layers for the file
- Finds it
- Copies it to the TWL

Take care of your container files.

Demo

Modify our run statement

- To mount a new volume
- Magically re-execute our config tasks

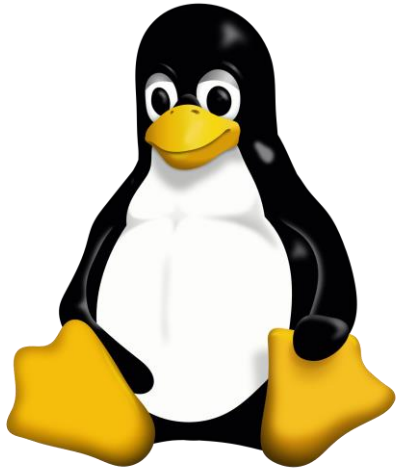
Poke around and look at the results

What they mean

- For container persistence

Wrapping up Jenkins State

The Docker Host isn't exactly Windows...



It's the Linux VM

Put your files where they belong

Summary

Why NOT build this stuff containerized?

- I don't see any good reason

Computers were once uniprocess

- Which required root permissions

Just like we moved away from that...

We're moving towards application isolation with containers

This is the new world

Get with it or get run over