Windows Containers

@CaptainShmaser http://blogs.blackmarble.co.uk/blogs/rviglianisi Black Marble

Riccardo Viglianisi

Devops Engineer/Software Developer

Microsoft Azure

Windows

Windows Phone

Visual Studio

Dynamics CRM

BizTalk

SharePoint

Kinect for Windows

black marble

Microsoft Partner

Gold Application Lifecycle Management
Gold Application Development
Gold Application Integration
Gold Cloud Platform
Gold Collaboration and Content
Gold Devices and Deployment
Gold Intelligent Systems
Silver Data Analytics
Silver Data Platform

Microsoft Partner of the Year
2015 Winner
Public Sector: Public Safety and National Security
Windows 8 Custom App Developer

The Agenda

Q: Why are we all here today

A: Dev Ops

Q: What am I doing here.

A: To talk about Docker

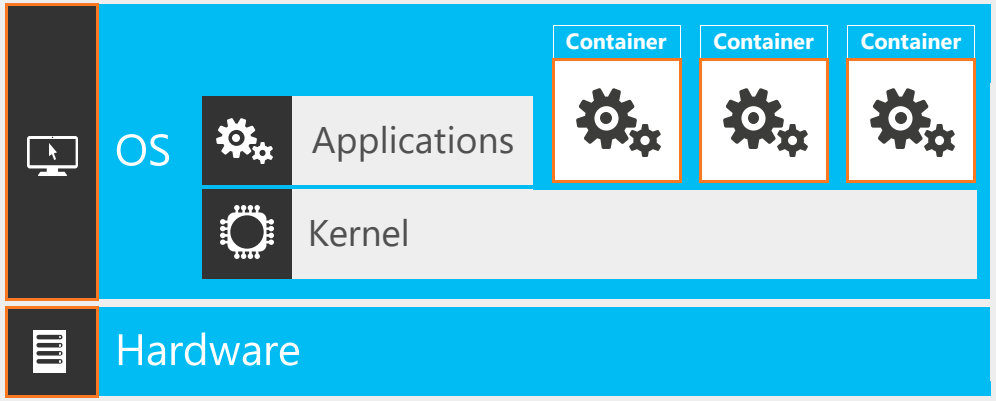
Docker Basics Crash Course

Really short story…

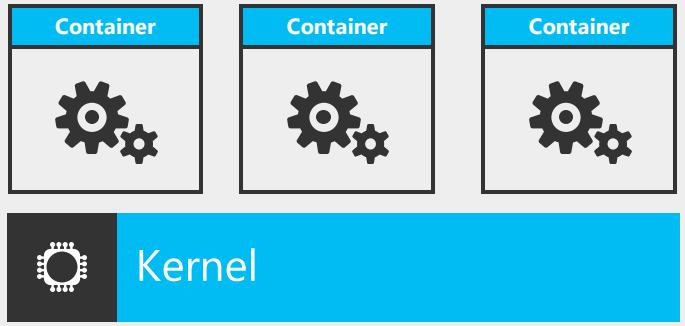
Google/Bing Search - “Docker is an open-source project that automates the deployment of Linux applications inside software containers. Docker provides an additional layer of abstraction and automation of operating-system-level virtualization on Linux.”

The Anatomy of Containers and VMs

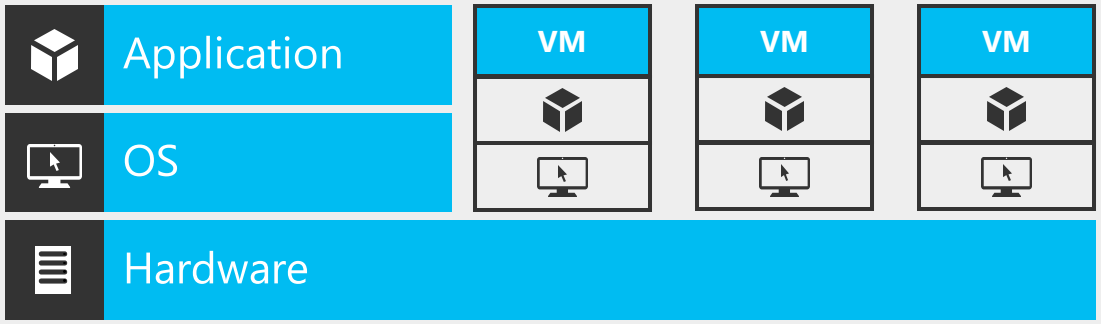
Containers = Operating system virtualization



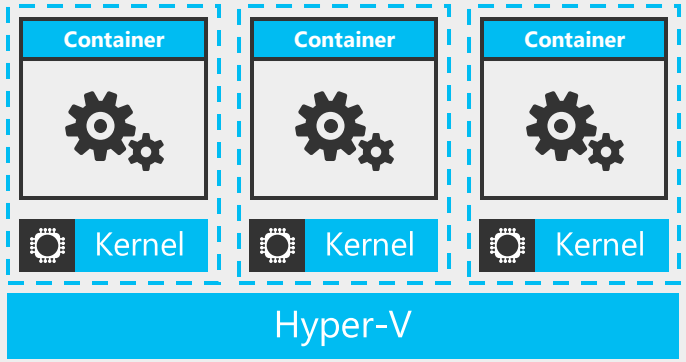
Windows Server containers
Maximum speed and density



Traditional virtual machines = hardware virtualization

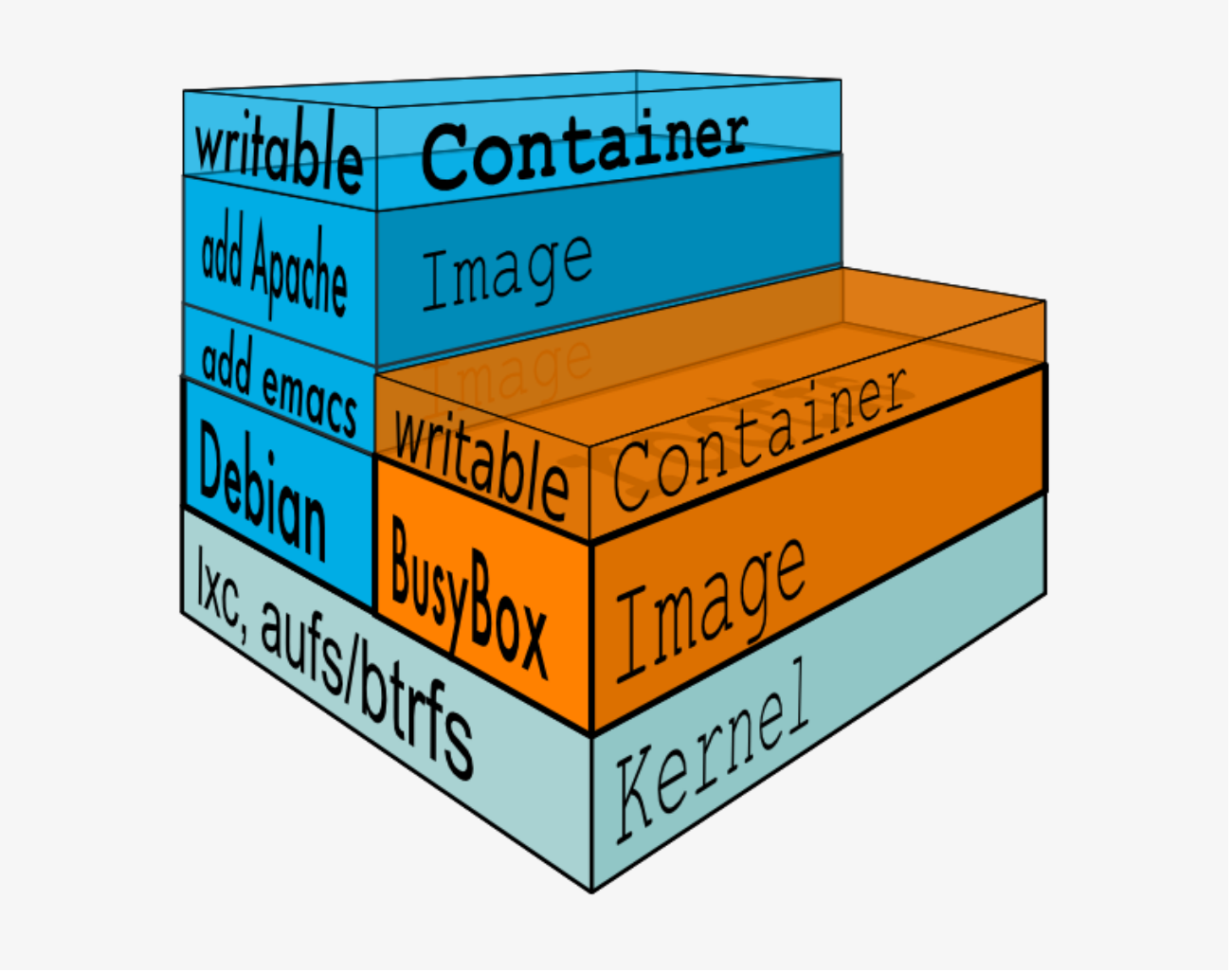


Hyper-V containers
Isolation plus performance



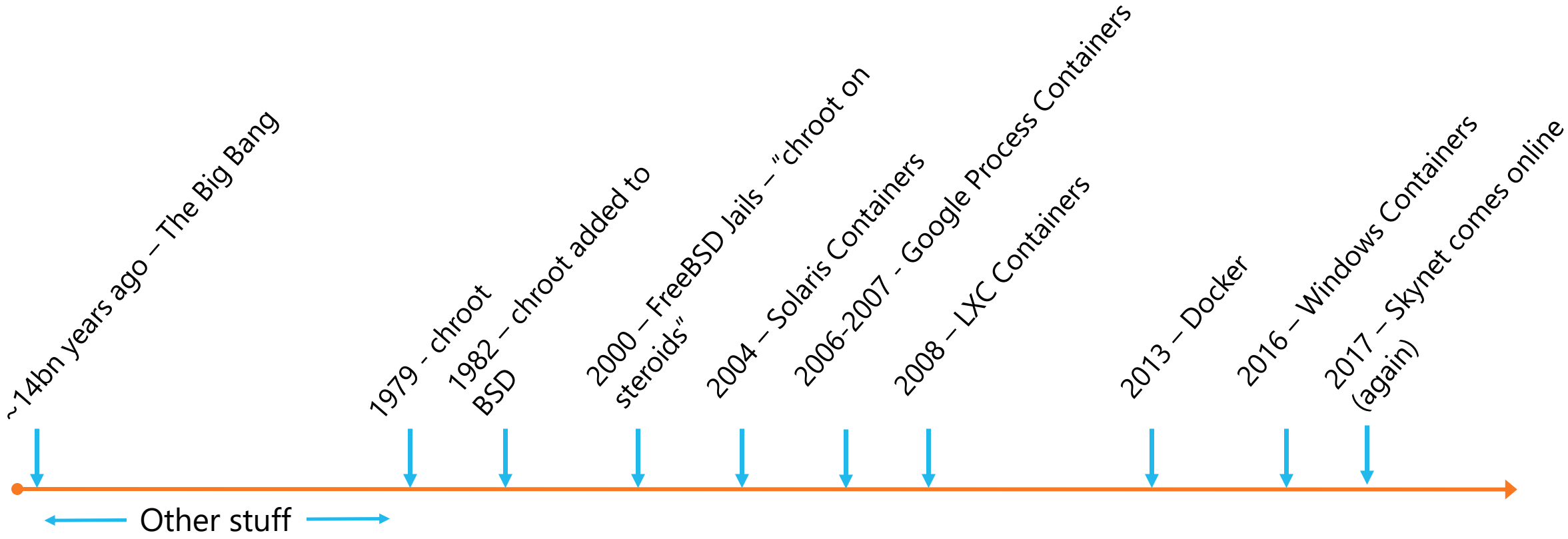
The Anatomy of a Docker Image

- Images are used to make Containers.
- They're built by Docker Files, a manifest of sorts.
- As the graphic shows they're build in layers, dictated by their Docker file
- Images can be stored in a repo (a bit like Git) and instantiated by pulling a copy of the Image via the Docker Engine.



How we got here: a brief history of containers

Not as new as you think.



In Summary

Super light weight virtualization technology.

Allows “containers” to run application in isolation from one another.

Promises scalability (rapid).

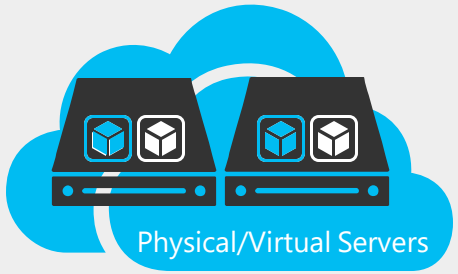
Containers are layered like a Scooby Doo sandwiches.

Historically a Linux/Unix only thing….until 2015/2016

The Docker Workflow

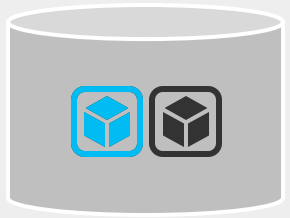
How it's advertised

The idealistic way of developers working with containers.



Operations collaborates with **developers** to provide app metrics and insights.

Developers build, test and update apps in containers, using development environment; i.e., Visual Studio.



Developers push containers to central repository.

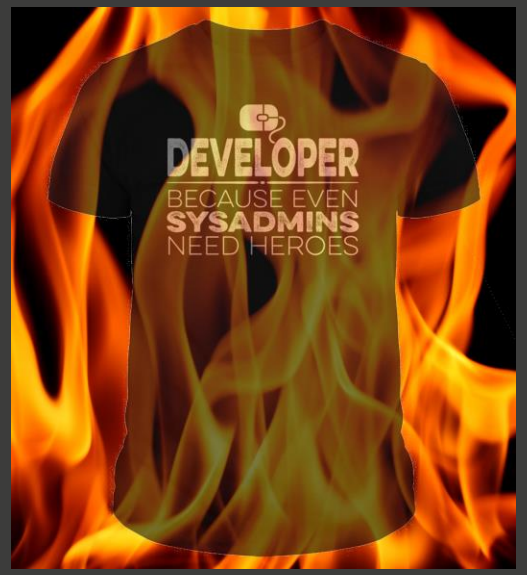
Operations automates deployment and monitors deployed apps from central repository.


DEVELOPER
BECAUSE EVEN
SYSADMINS
NEED HEROES

“Who the hell uses Public Images from Docker?

They could be made by Russian Hackers for all you know!”

Downfall Docker Parody https://www.youtube.com/watch?v=PivpCKEiQOQ



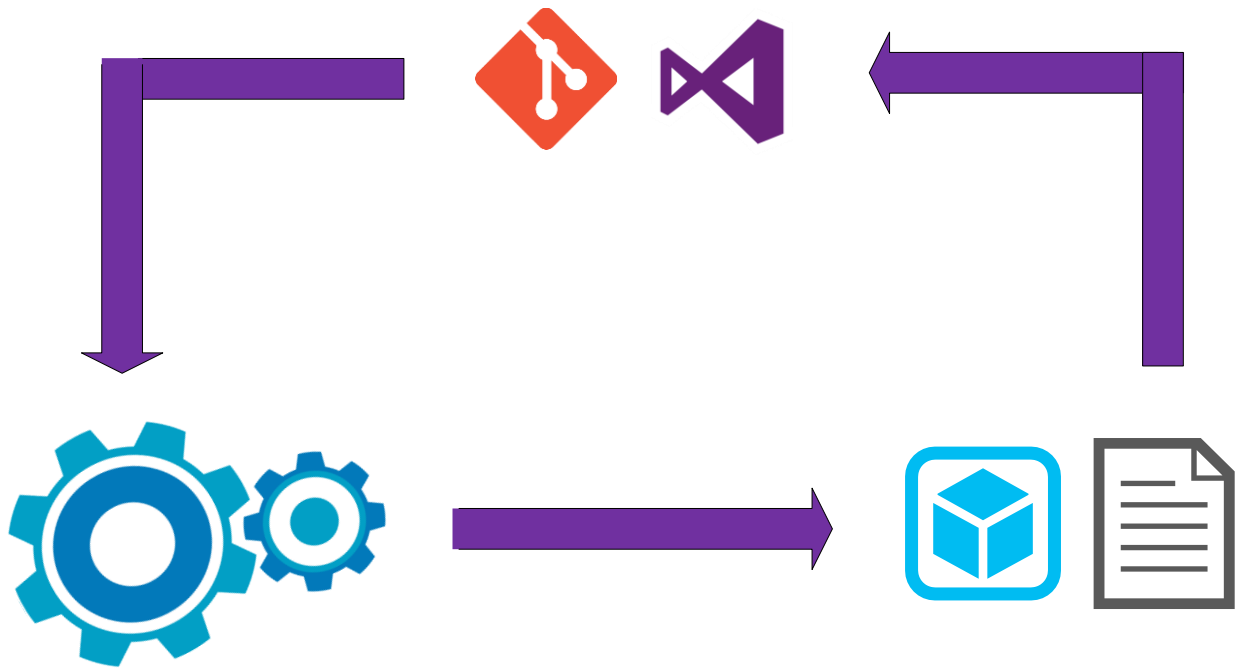
Demo – Building a Docker Image you can hand to your IT team

And not be shot

The Docker Workflow 2.0

The way it should be done.

- Images should not built in isolation (on a developer machine) but created and accessed via private repos
- Images should be subject to versioning
- Images should be built via a standard CI, then release pipeline before being utilised in production.



Demo- Deploying your IT Approved Docker image in a consistent fashion

Behold the Power of ARM!

Docker Applications

What would you use it for?

Build Agent

Strengths

- Leverages Container scalability
- Containers are sterile and clean

Limitations

- Software requirements can be tricky if required assemblies cannot be NuGet packages
- Older SDKS will bloat out your Docker image, reduces speed Container scalability

Micro-Services

Strengths

- Best use of container scalability
- Leverage Container Density

Limitations

- Requires orchestrator to manage efficiently at large scale e.g. Docker Swarm

Web Service

Strengths

- It's another example of Scaling being great.
- No configuration needed, deployment/configuration is baked into the Image.
- Easy to replace
- Easy to replicate (in load balancing scenarios)

Limitations

- Nothing worth noting.

App-Tier Host e.g SQL

Strengths

- Effectively creates a headless server, no GUI, good for security.
- Smaller resource foot print, because you're not even running a full OS.
- No "Live Migration"

Limitations

- No "Live Migration"
- Less secure than a traditional VM, no virtualized/physical hardware security features.
- Large images
- App tiers are typically managed via GUI's, best hope you've got good remote tool support.

Summary

It’s a great technology but…..

Virtual Machines are not going anywhere.

Containers in Production for windows is still Niche

That said…….it’s something we should keep track of, and look for opportunities to use where possible IF it is the best tool for the job.


That's all Folks!

blogs.blackmarble.co.uk

+44 (0)1274 300175

@blackmarble

Black Marble Ltd.

Black Marble

black marble

Microsoft Partner

Gold Application Lifecycle Management
Gold Application Development
Gold Application Integration
Gold Cloud Platform
Gold Collaboration and Content
Gold Devices and Deployment
Gold Intelligent Systems
Silver Data Analytics
Silver Data Platform