



Containers

Miguel Caldas
Principal Technical Evangelist

Works on my machine

Set of IT Problems **forever**
Patches break installations
Deployment binders
Lead to shadow IT, tension between
dev/ops
Etc

The cloud has changed expectations



Availability
100% Uptime



Hyper-scale
From startup to
enterprise



Agility
Deliver just in
time speed

What problems do containers solve?

Containers deliver speed, flexibility, and savings



Availability

62%

Report reduction in MTTR

10X

Cost reduction in maintaining
existing applications



Hyper-scale

41%

Move workloads across
private/public clouds

Eliminate

"works on my machine" issues



Agility

13X

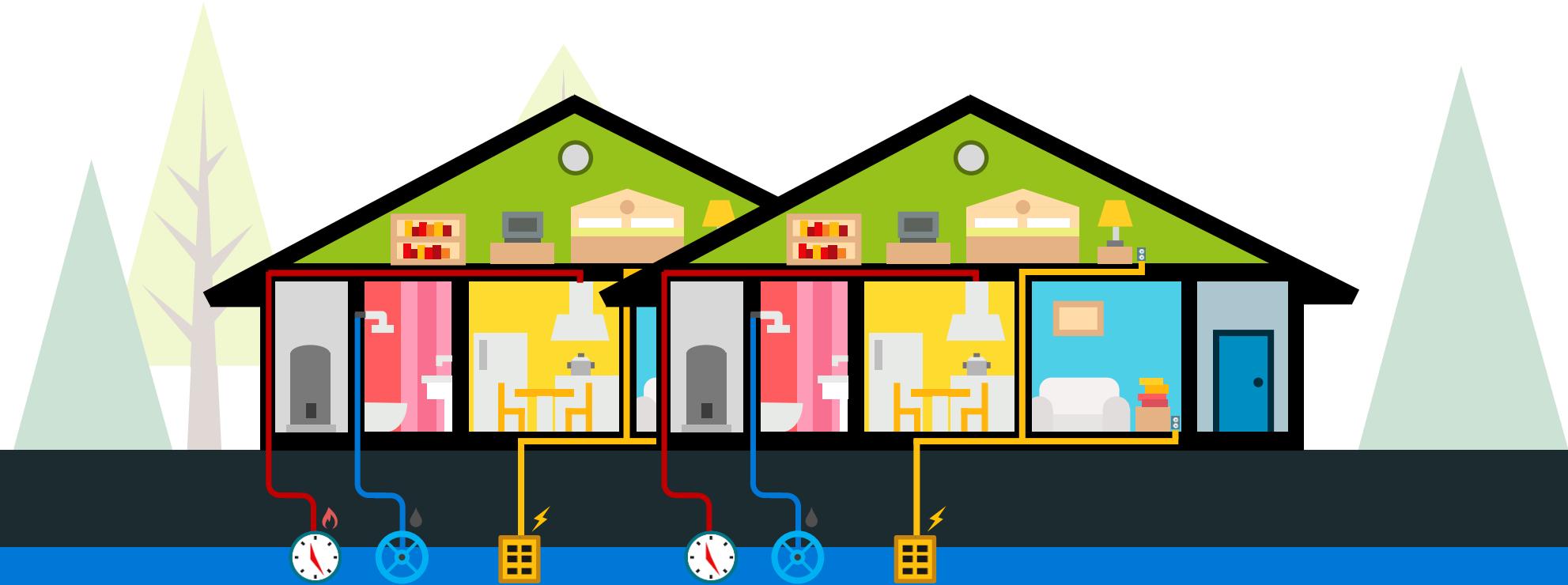
More software releases

65%

Reduction in developer
onboarding time

What is a container?

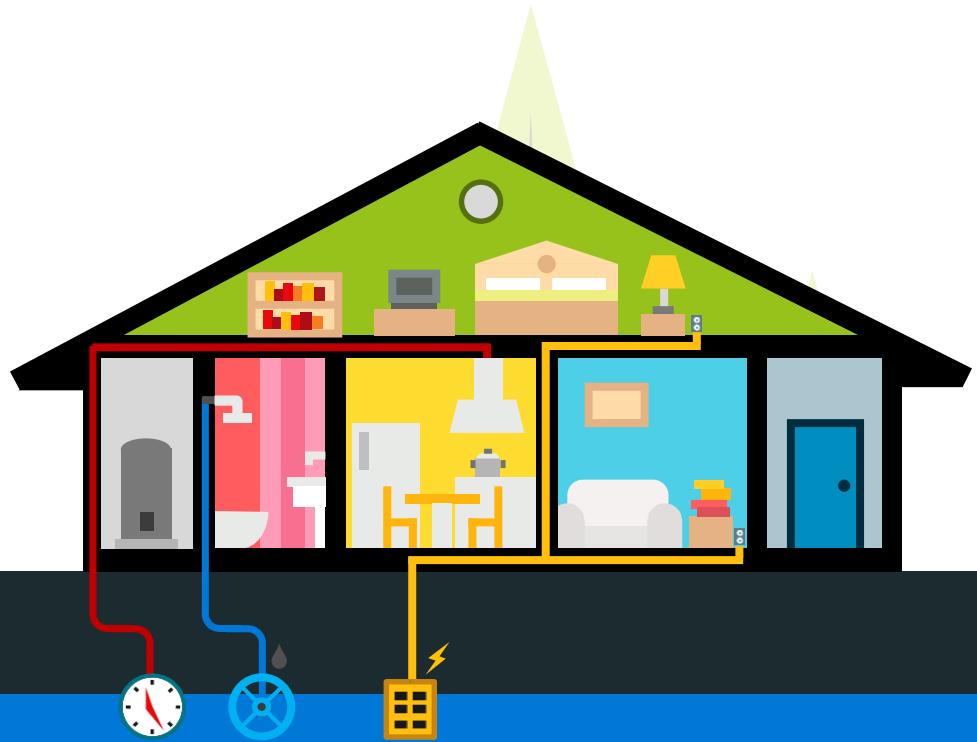
Virtual Machine (VM) vs. Container



Virtual Machine (VM) vs. Container

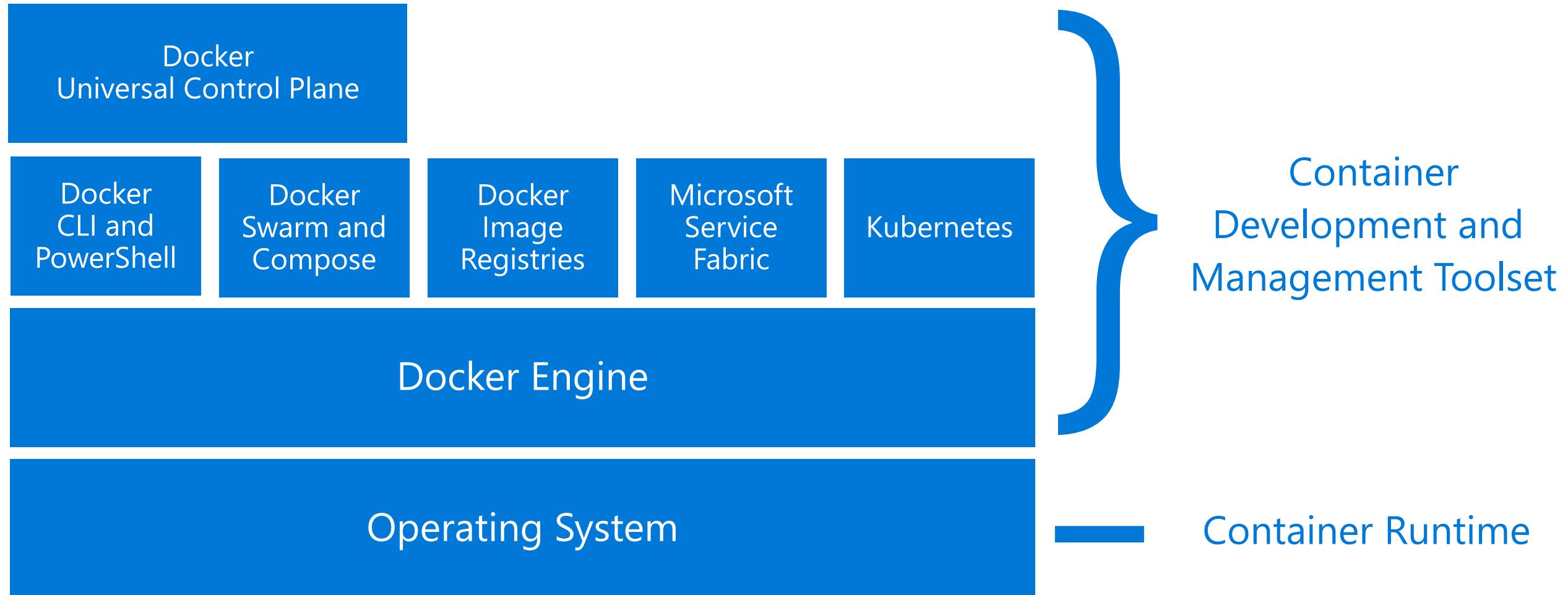


Virtual Machine (VM) vs. Container

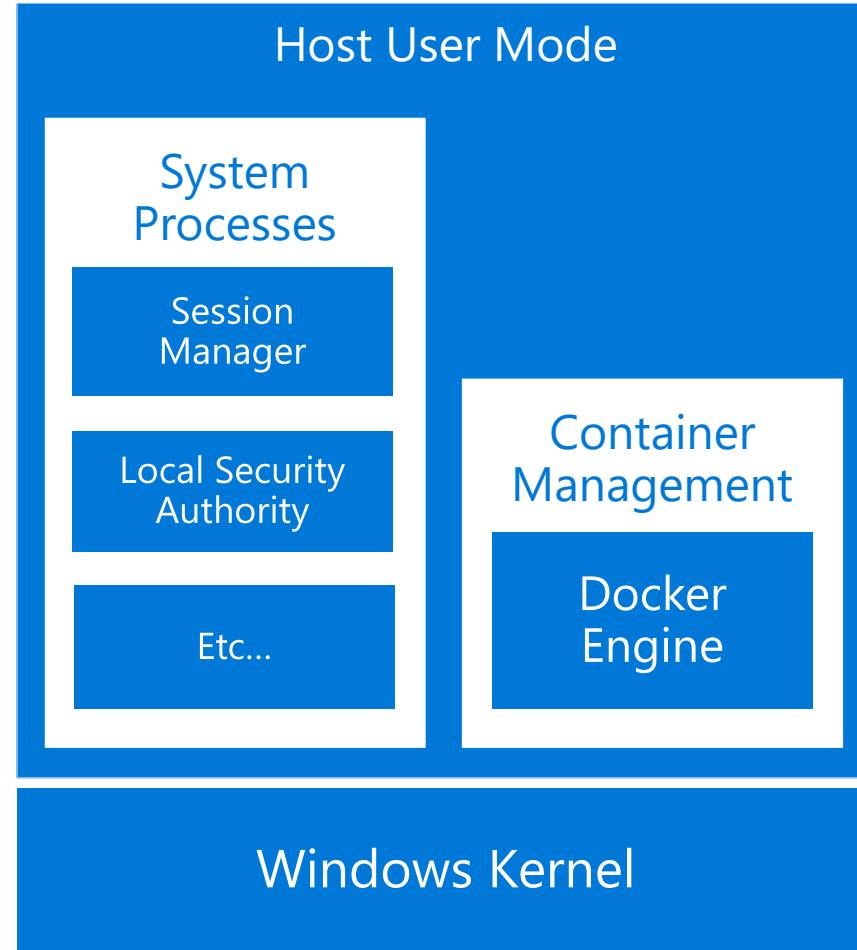


How do containers work?

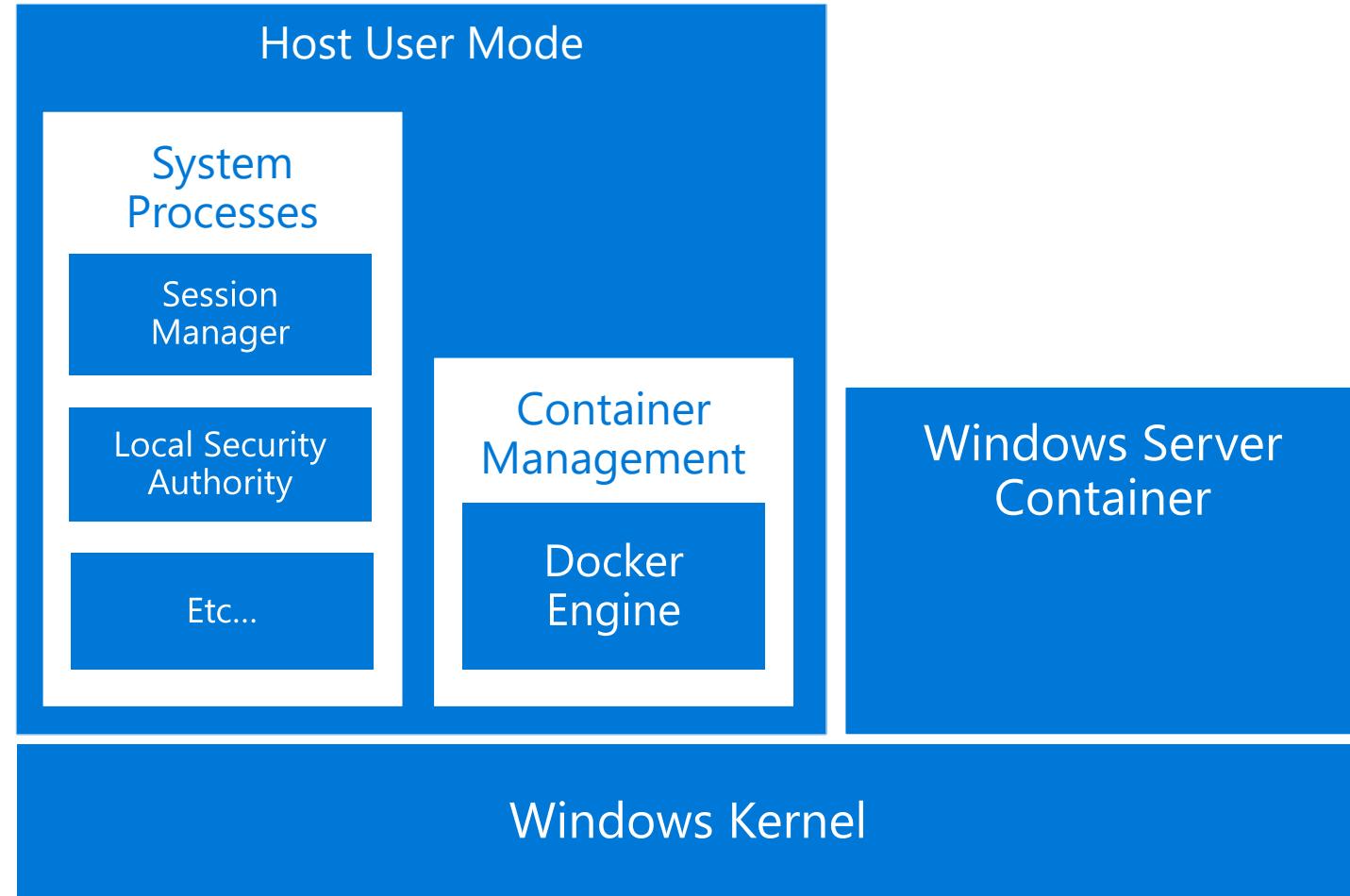
High Level Architecture



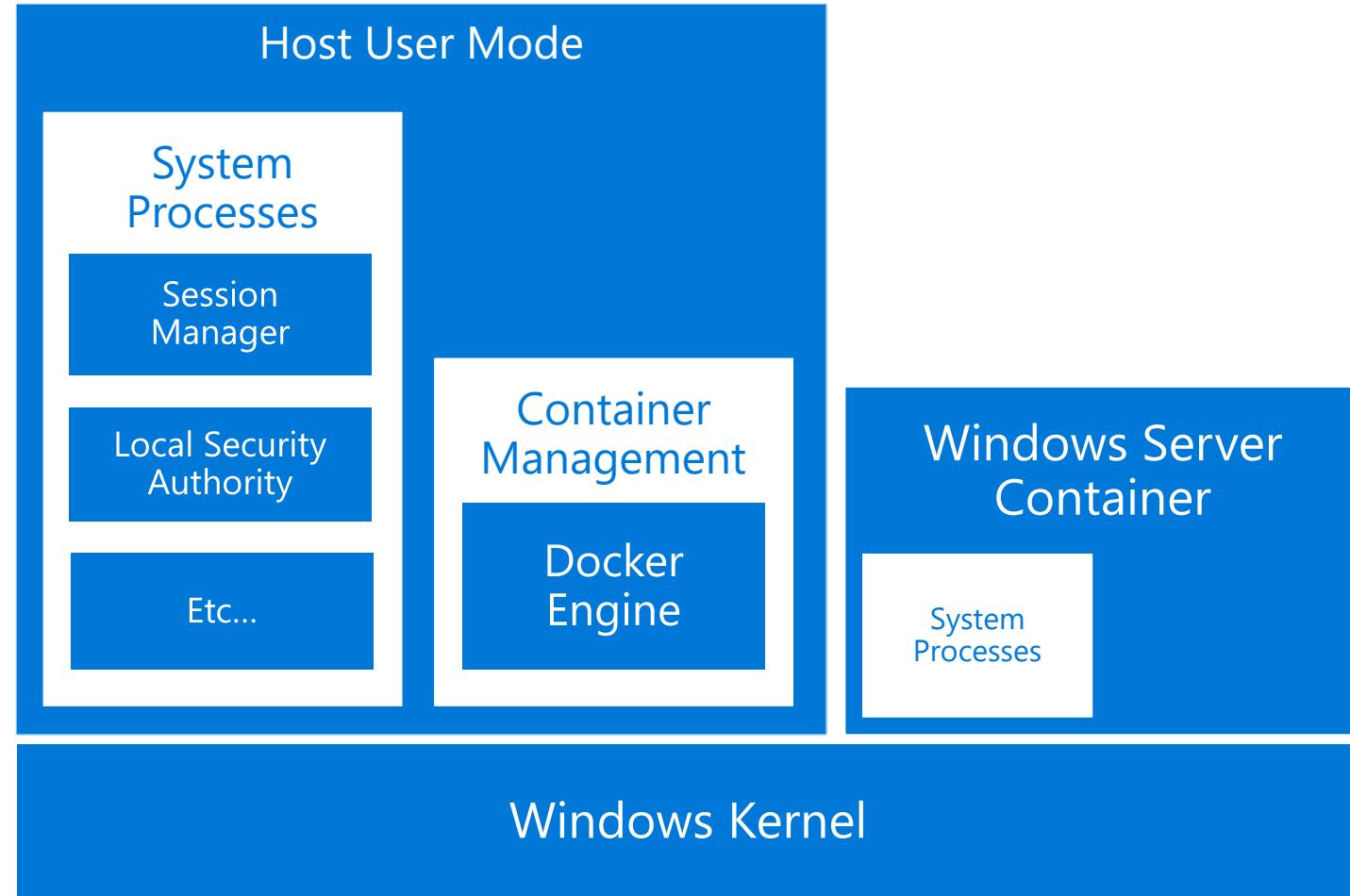
Windows Containers



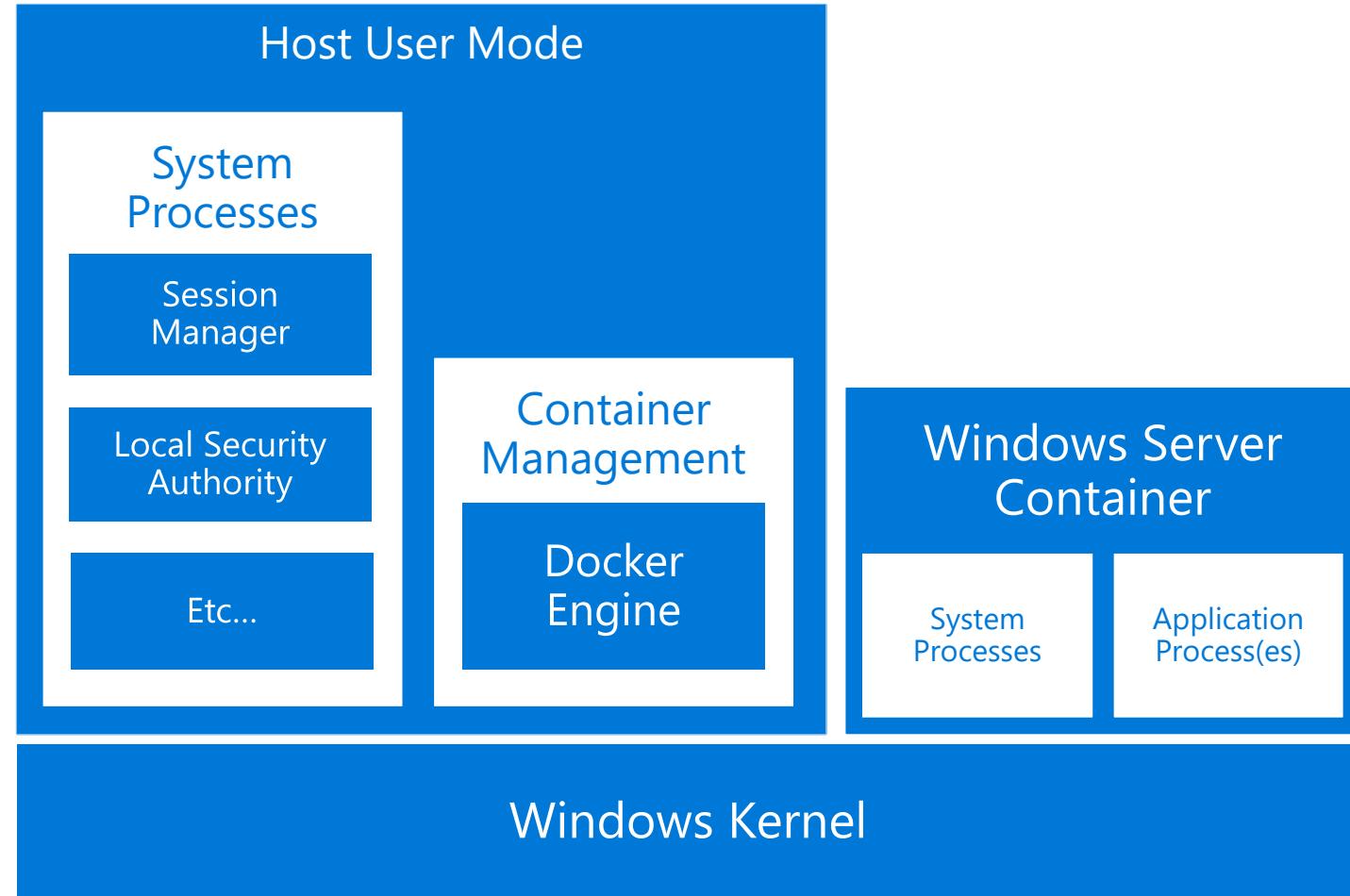
Windows Containers



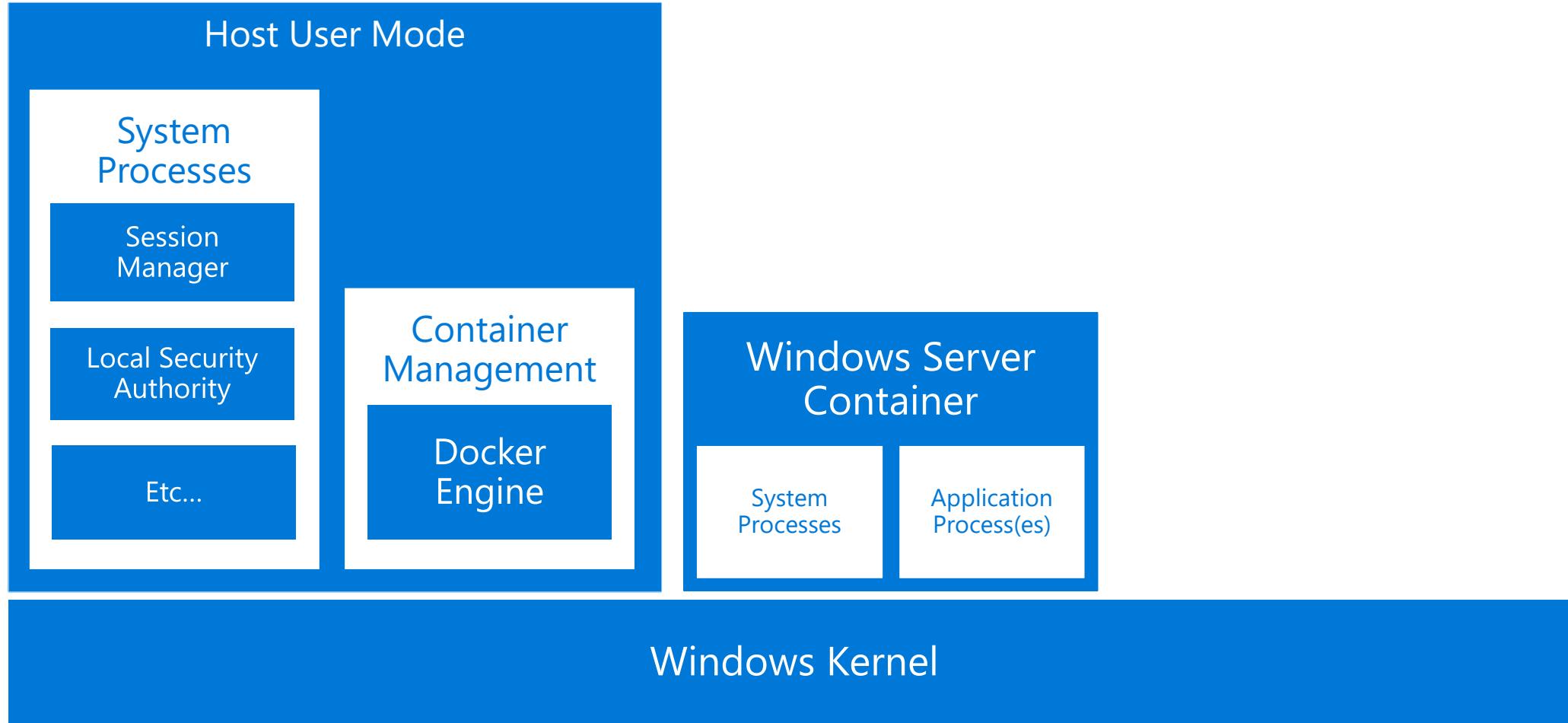
Windows Containers



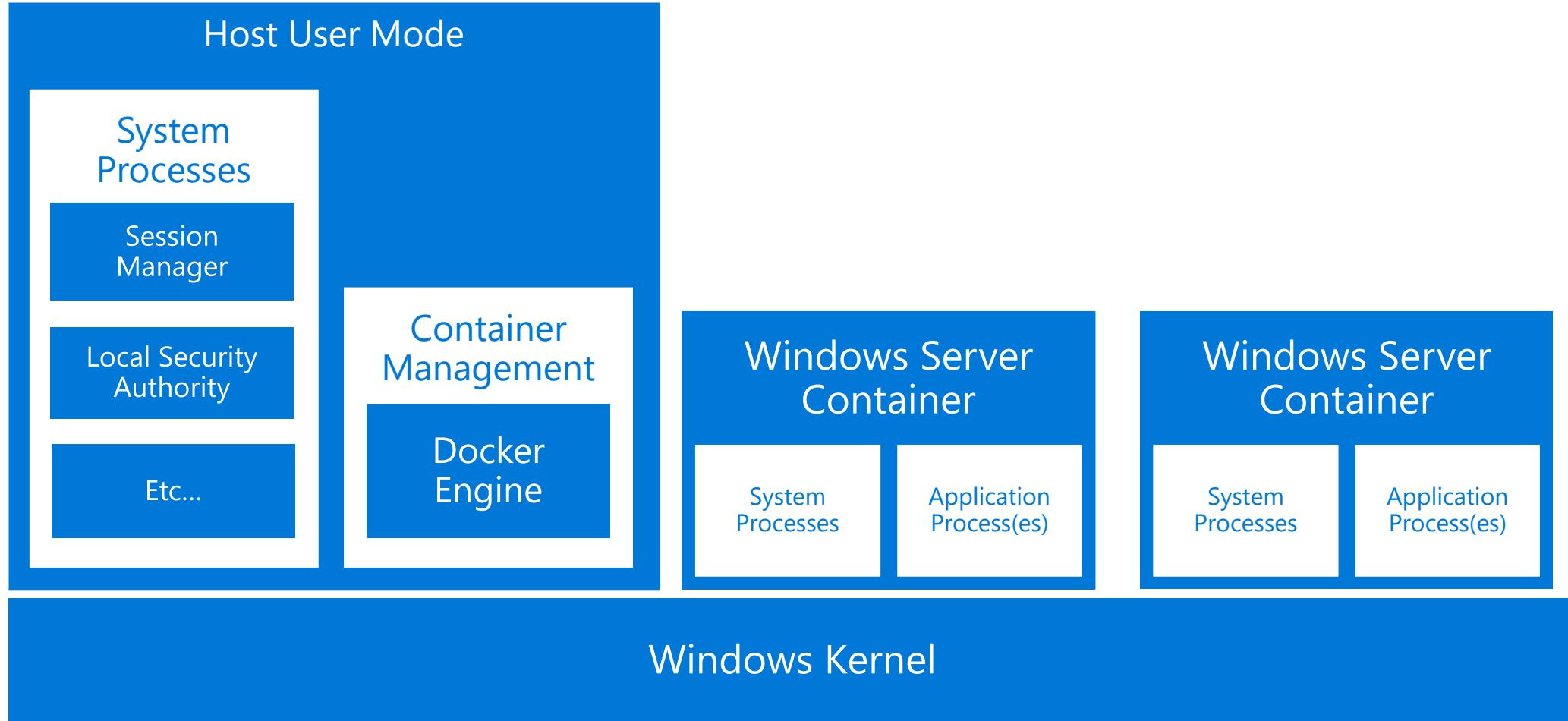
Windows Containers



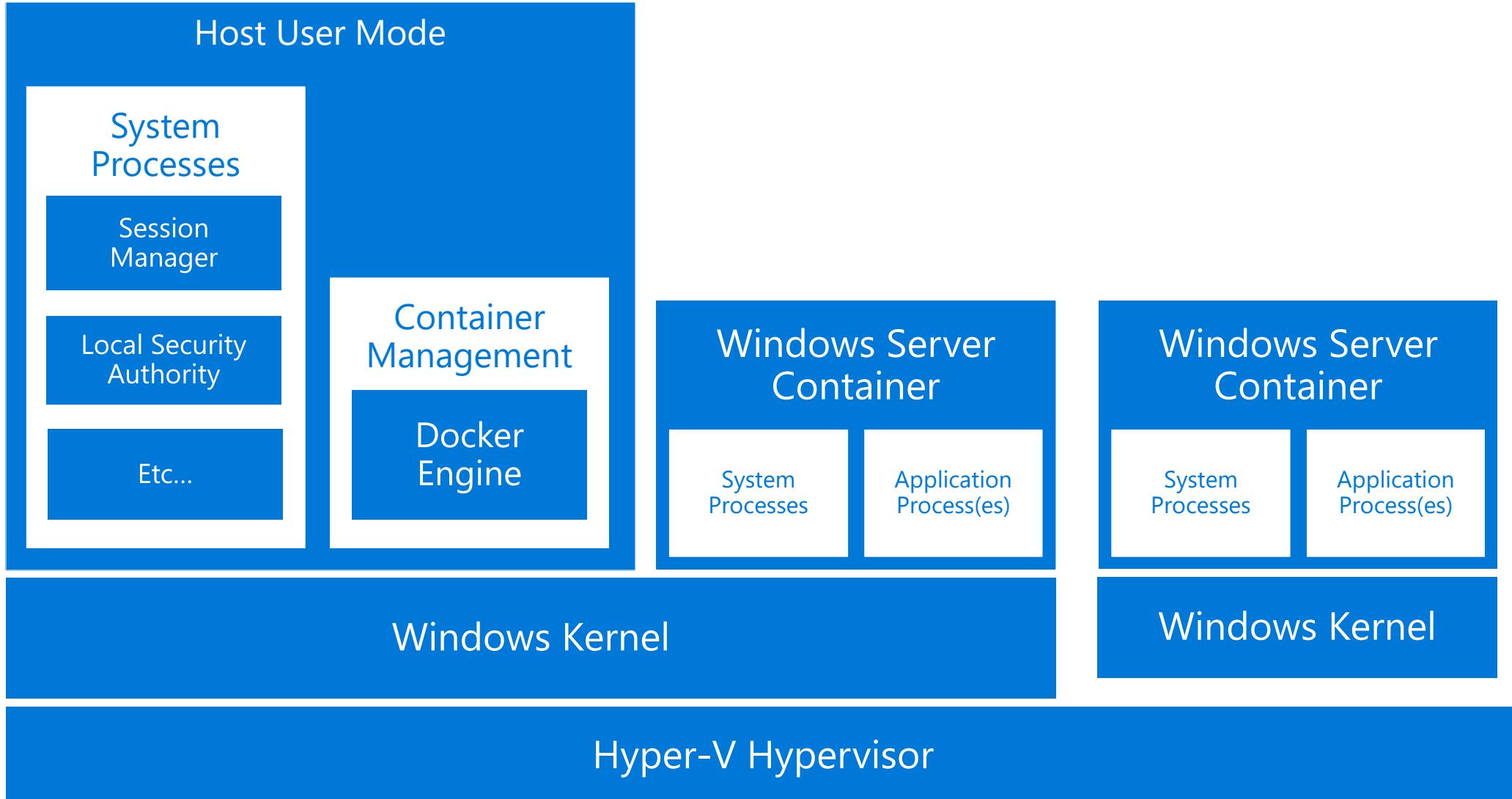
Windows Containers



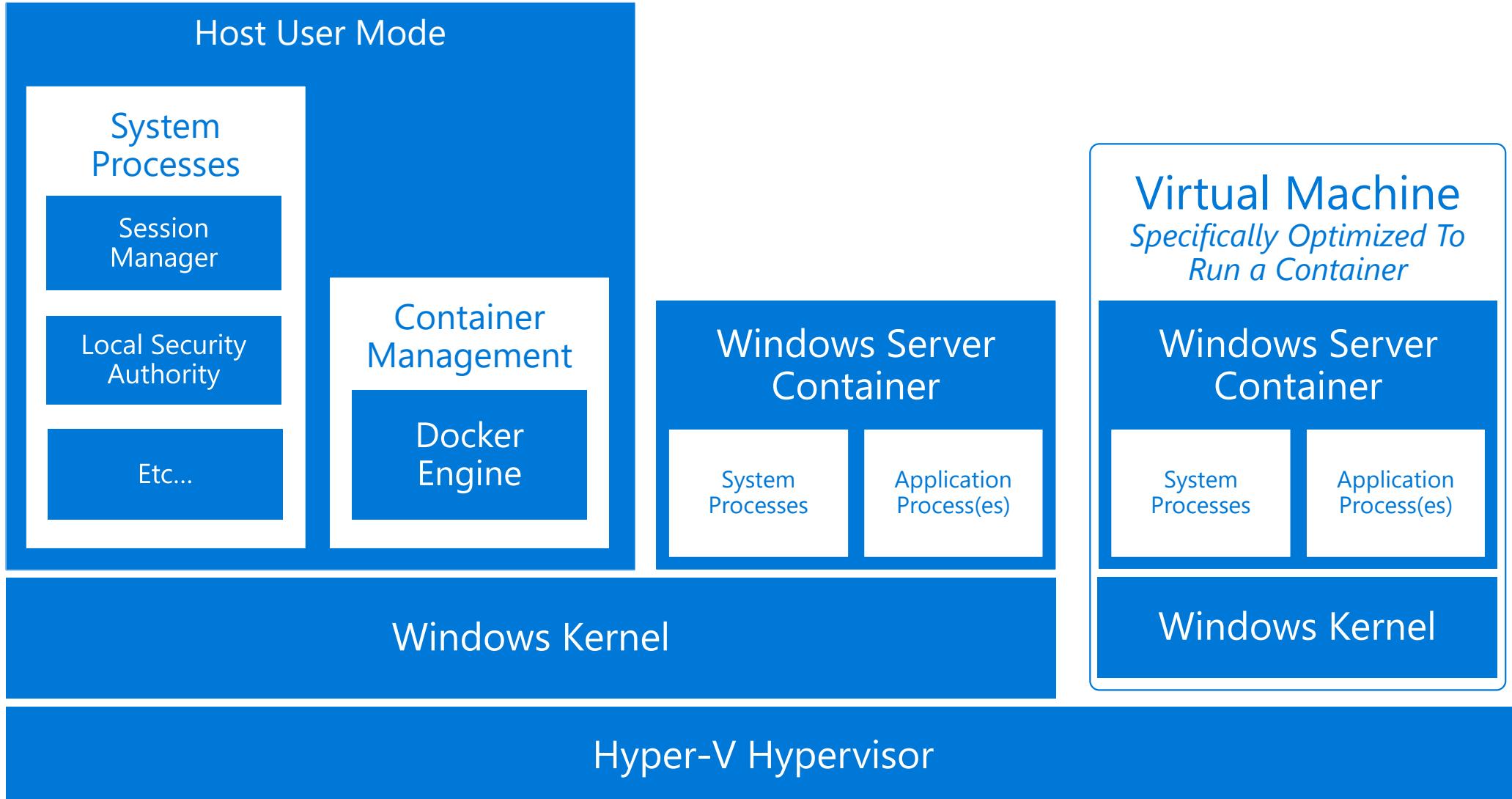
Windows Containers



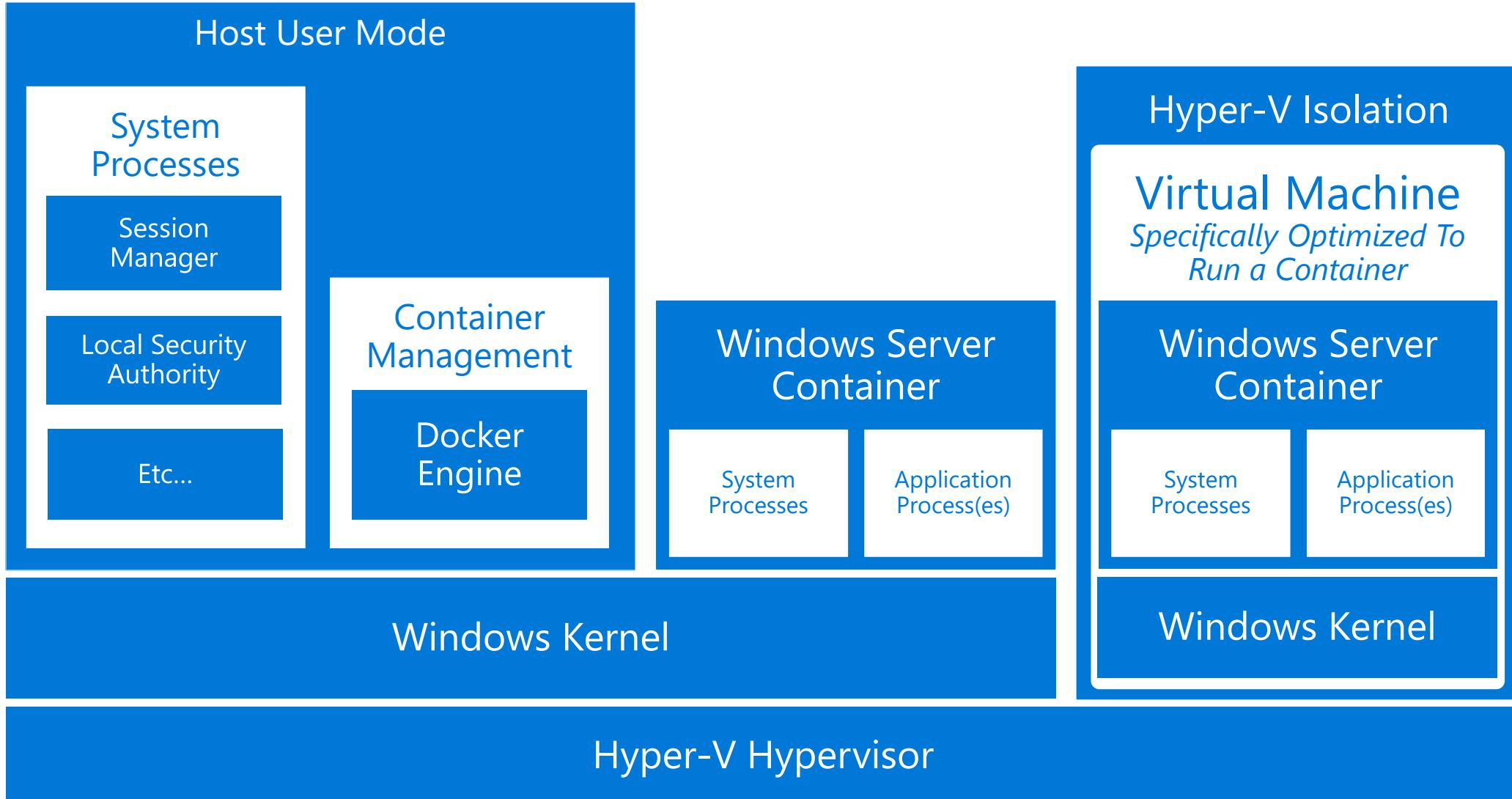
Windows Containers



Windows Containers



Windows Containers



Startup Performance

NodeJS with Windows Server Core

Windows Server Container
~1 second

With Hyper-V Isolation
~3.3 seconds

A virtual machine takes ~**5 seconds to over a min**

*Includes initial boot provisioning phase (out-of-box setup etc...)

**Startup time after initial container start.

Testing performed on HP ProLiant SL250s Gen8, E5-2600, 2 Socket, 8 Core, 128GB RAM, HP SATA SSD - results may vary based on hardware and software configurations.

Startup Performance

NodeJS with Nano Server

Windows Server Container
Under 600 Milliseconds!

With Hyper-V Isolation
~1.75 seconds

A virtual machine takes ~**3 seconds**

NodeJS with Windows Server Core

Windows Server Container
~1 second

With Hyper-V Isolation
~3.3 seconds

A virtual machine takes ~**5 seconds to over a min**

Density

NodeJS with Windows Server Core

Windows Server Container

First Container ~**150MB**
Additional Containers ~**75MB**

With Hyper-V Isolation

First Container ~**555MB**
Additional Containers ~**280MB**

Density

NodeJS with Nano Server

Windows Server Container

First Container ~**120MB**
Additional Containers ~**75MB**

With Hyper-V Isolation

First Container ~**340MB**
Additional Containers ~**150MB**

NodeJS with Windows Server Core

Windows Server Container

First Container ~**150MB**
Additional Containers ~**75MB**

With Hyper-V Isolation

First Container ~**555MB**
Additional Containers ~**280MB**

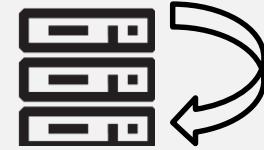
Where should I start?

One platform, one journey



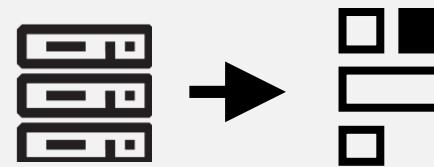
CONTAINERIZE TRADITIONAL APPLICATIONS

Containerize for security, portability and efficiency



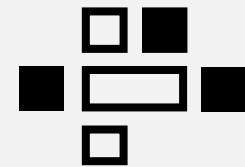
TRANSFORM MONOLITHIC TO MICROSERVICES

Look for shared services to transform



ACCELERATE NEW APPLICATIONS

Agile cloud native app development



Why containerize traditional Windows apps?

PORTABILITY, SECURITY AND COST REDUCTION

- Accelerate your public and hybrid cloud strategy
- Gain applications portability, agility, and control
- Reduce infrastructure and management costs for Windows 2003, 2008, and 2012 Windows applications

COSTS FOR MANAGING TRADITIONAL APPLICATIONS



Top Development Challenges for Development Teams

65%

Traditional Application Maintenance

59%

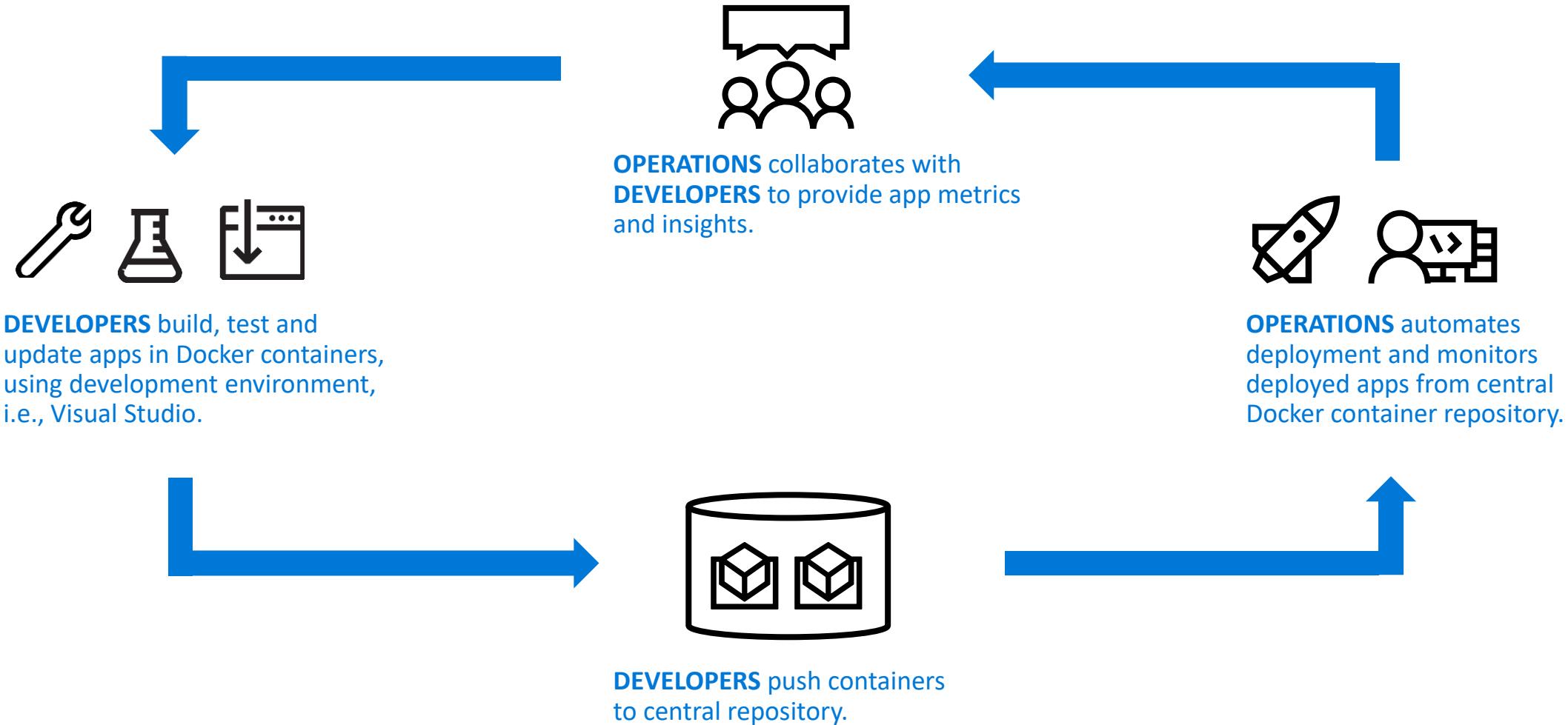
Inertia of Traditional App Infrastructure

“ Most of our application portfolio consists of older legacy applications that are cumbersome to update. By moving these applications into Windows Server Containers with Docker and embracing a microservices architecture, we can break these big applications apart and update the pieces independently. This will reduce customer downtime and increase business agility. ”

STEPHEN TARMEY
Chief Architect
Tyco International

Collaborative DevOps Approach to apps

Docker containers are central to process



How do I actually create a container?

Docker Run

Creates and Starts a New Container

Runtime options

Name (network name and management name)

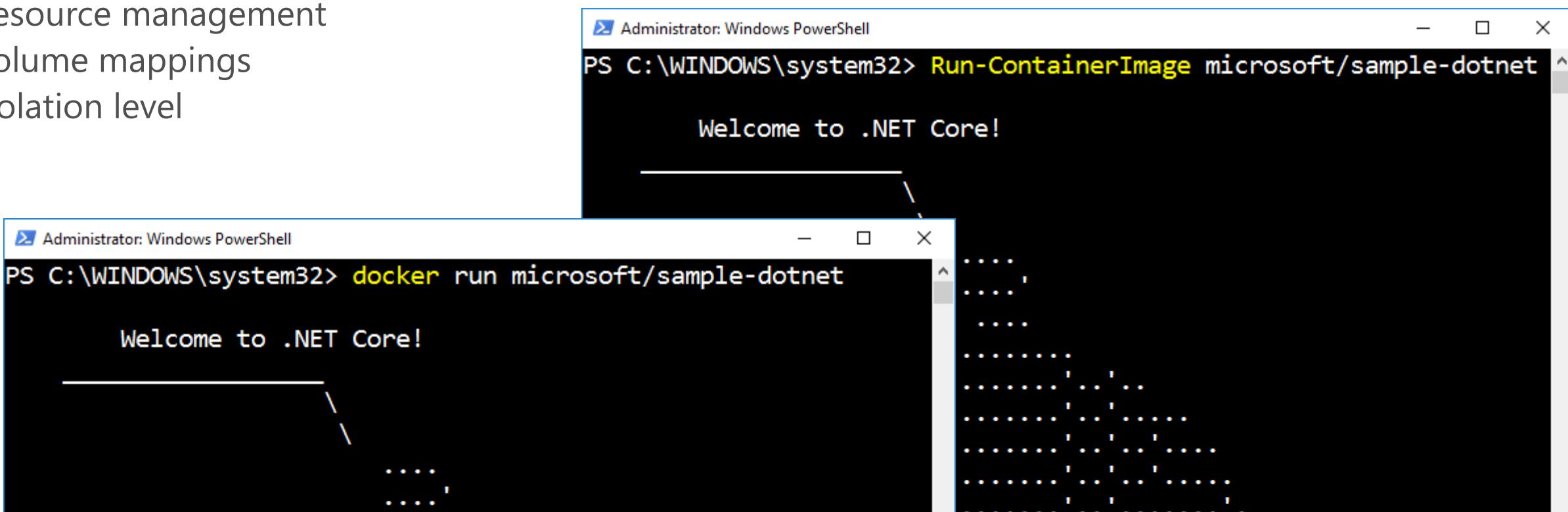
Interactive or Service

Network configuration

Resource management

Volume mappings

Isolation level



The image shows two side-by-side Windows PowerShell windows. Both are titled "Administrator: Windows PowerShell". The top window has a blue border and displays the command "PS C:\WINDOWS\system32> Run-ContainerImage microsoft/sample-dotnet" followed by the text "Welcome to .NET Core!". The bottom window has a black border and displays the command "PS C:\WINDOWS\system32> docker run microsoft/sample-dotnet" followed by the same "Welcome to .NET Core!" message. The background of the slide features a faint grid pattern.

```
Administrator: Windows PowerShell
PS C:\WINDOWS\system32> Run-ContainerImage microsoft/sample-dotnet
Welcome to .NET Core!

Administrator: Windows PowerShell
PS C:\WINDOWS\system32> docker run microsoft/sample-dotnet
Welcome to .NET Core!
```

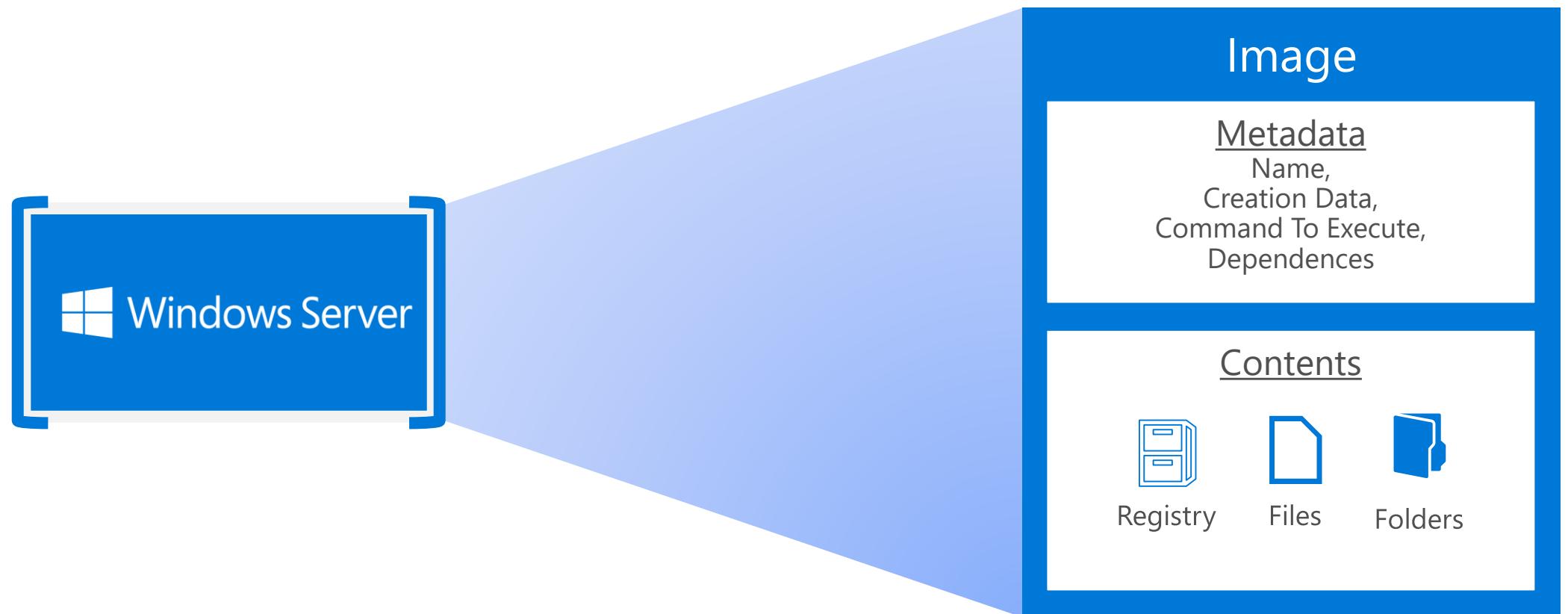
What is a container image?

Container Image

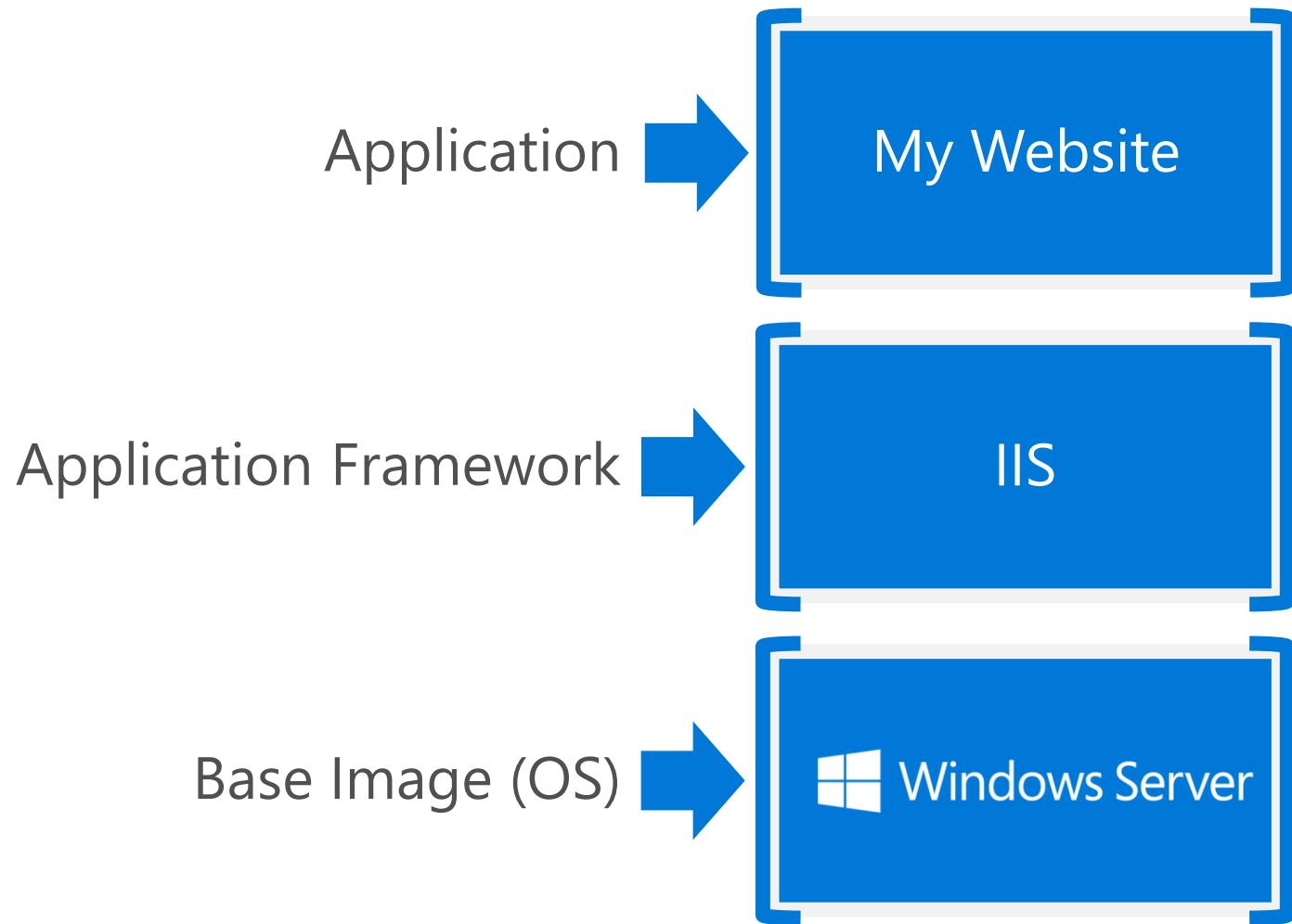
Analogous to a VHD and config file to a virtual machine

Created by running a container and capturing changes

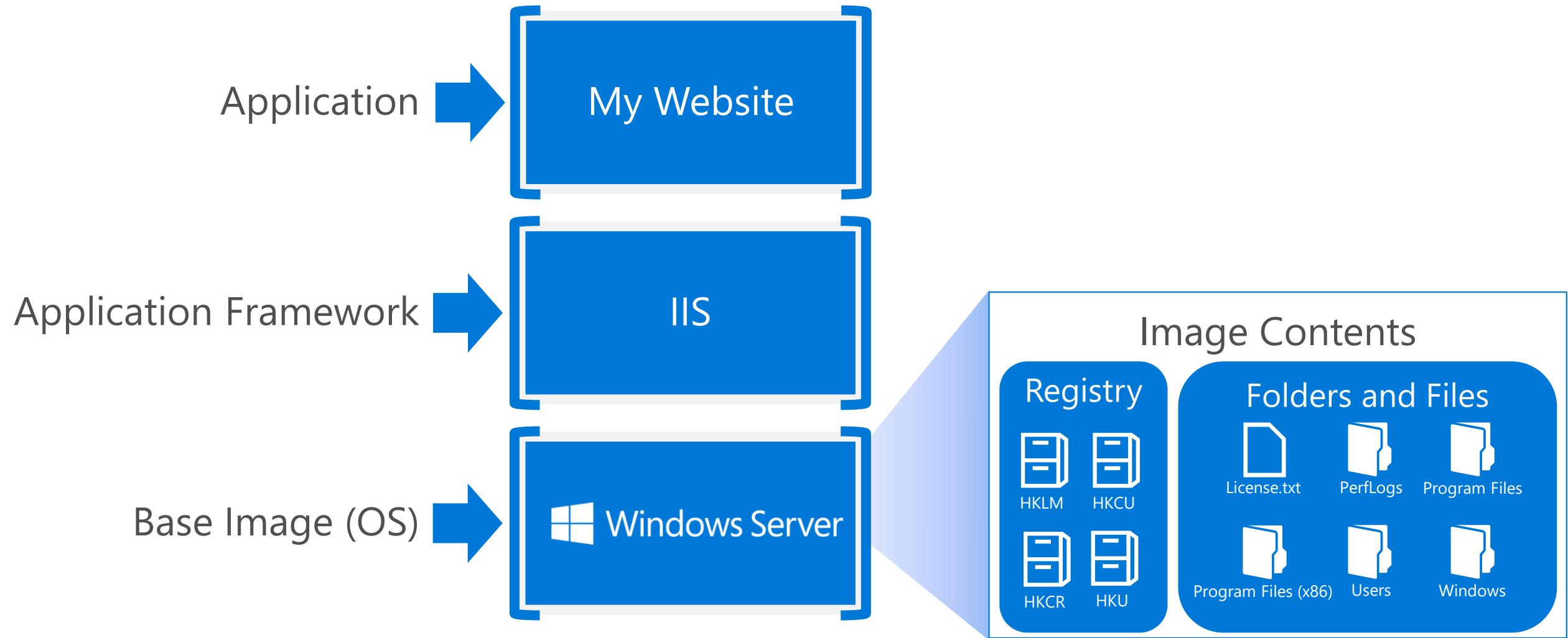
Changes include files and registry



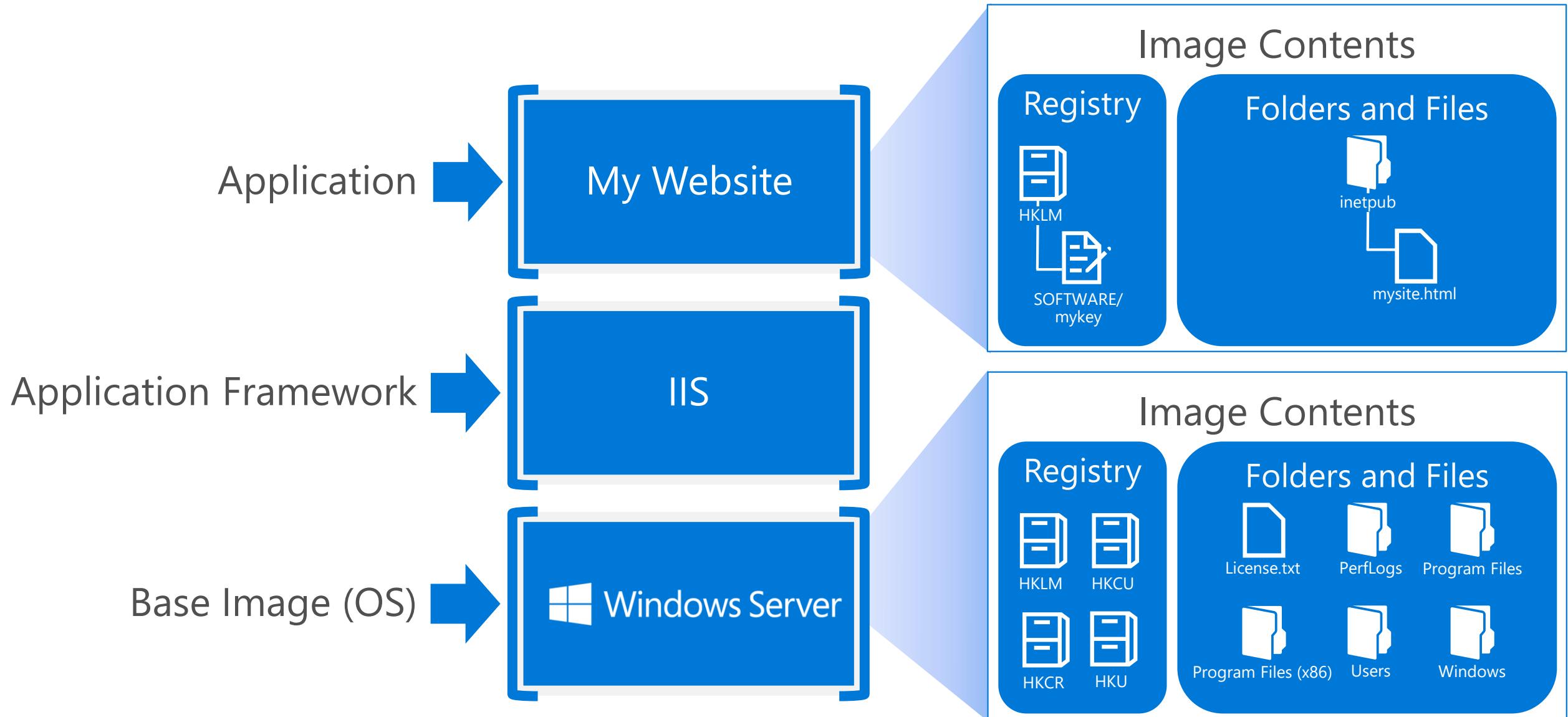
Container Images



Container Images



Container Images



Automated Image Building

Docker Build and Dockerfiles

Method for automated container image build

Consumed when running “docker build”

Caches unchanged commands

Integrates into Docker Hub

Examples

IIS

```
FROM windowsservercore  
RUN powershell –command Add-WindowsFeature Web-Server
```

Website

```
FROM iis  
ADD mysite.htm inetpub\mysite.htm
```

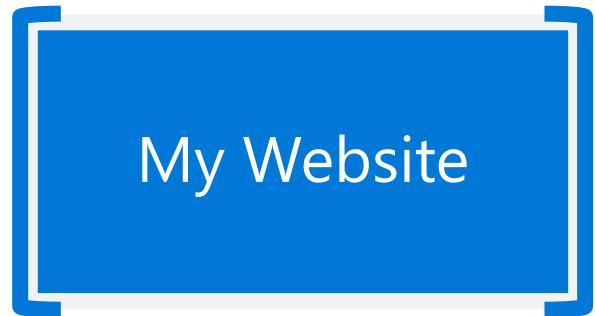


Image Registries

What is a registry?

Stores container images

Images are **Pushed** into a registry

Images are **Pulled** from a registry

Images are **Searched** for within a registry

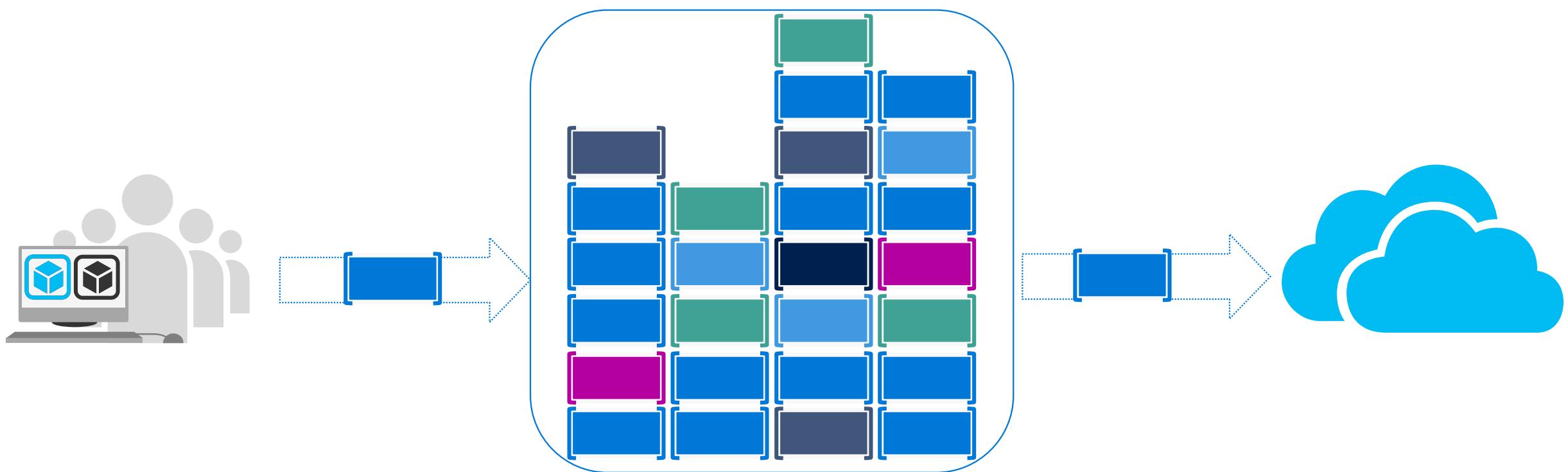


Image Registries

Docker Hub and Docker Store

Public, Official and Private image repositories

Granular access controls with organization support

Automated image build support

Docker Trusted Registry

Enterprise Grade Private Registries

Runs on your infrastructure (on-prem or cloud)

Active Directory and Role Based Access Controls

Azure Container Registry

Store and manage container images across Azure deployments

Maintain Windows and Linux container images

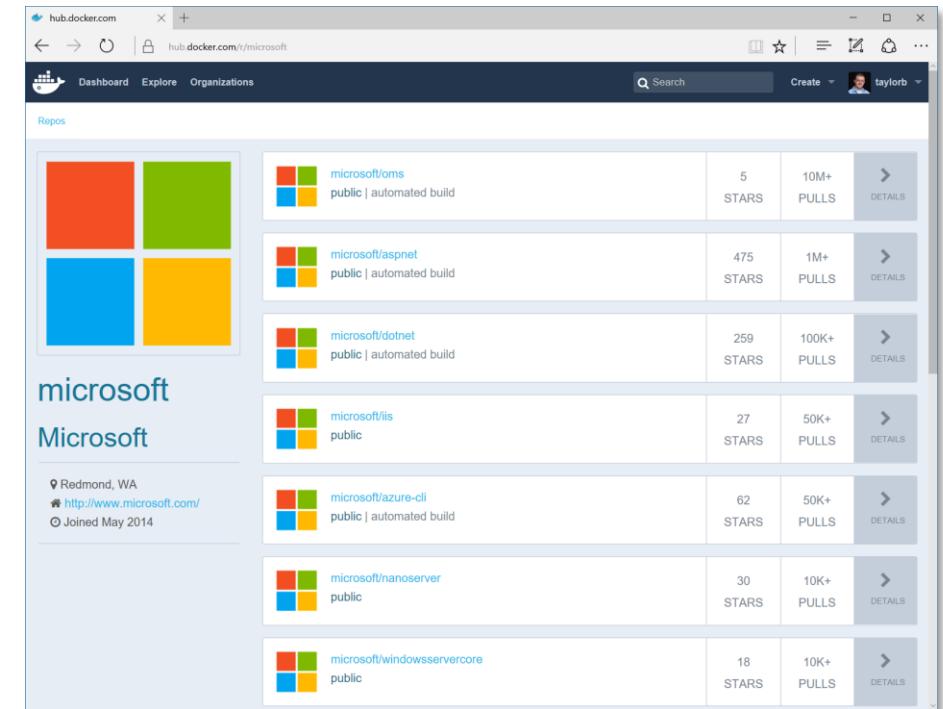
Same API and Tools as Docker Hub/Store/Registry

Docker Registry

Open source foundation of Hub and DTR

Runs on your infrastructure (on-prem or cloud) as a container

<https://docs.docker.com/registry> and or <https://github.com/docker/distribution>



What about...

Licensing



Windows Server 2016 feature differentiation and core-based pricing		
Feature	Datacenter	Standard
Core functionality of Windows Server	•	•
OSEs / Hyper-V containers	Unlimited	2
Windows Server containers	Unlimited	Unlimited
Host Guardian Service	•	•
Nano Server*	•	•
Storage features including Storage Spaces Direct and Storage Replica	•	
Shielded Virtual Machines	•	
Networking stack	•	
Core-based pricing**	\$6,155	\$882

*Software Assurance is required to deploy and operate Nano Server in production.

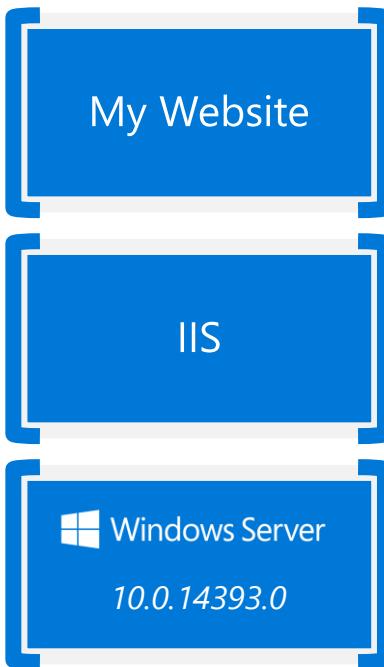
**Pricing for Open (NL) ERP license for 16 core licenses. Actual customer prices may vary.

Patching and Updates

Update Container OS Image

Pull updated base image

Rebuild containers using dockerfiles

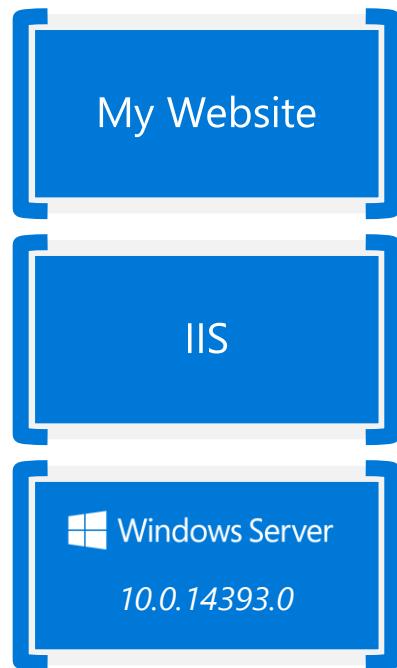


Patching and Updates

Update Container OS Image

Pull updated base image

Rebuild containers using dockerfiles

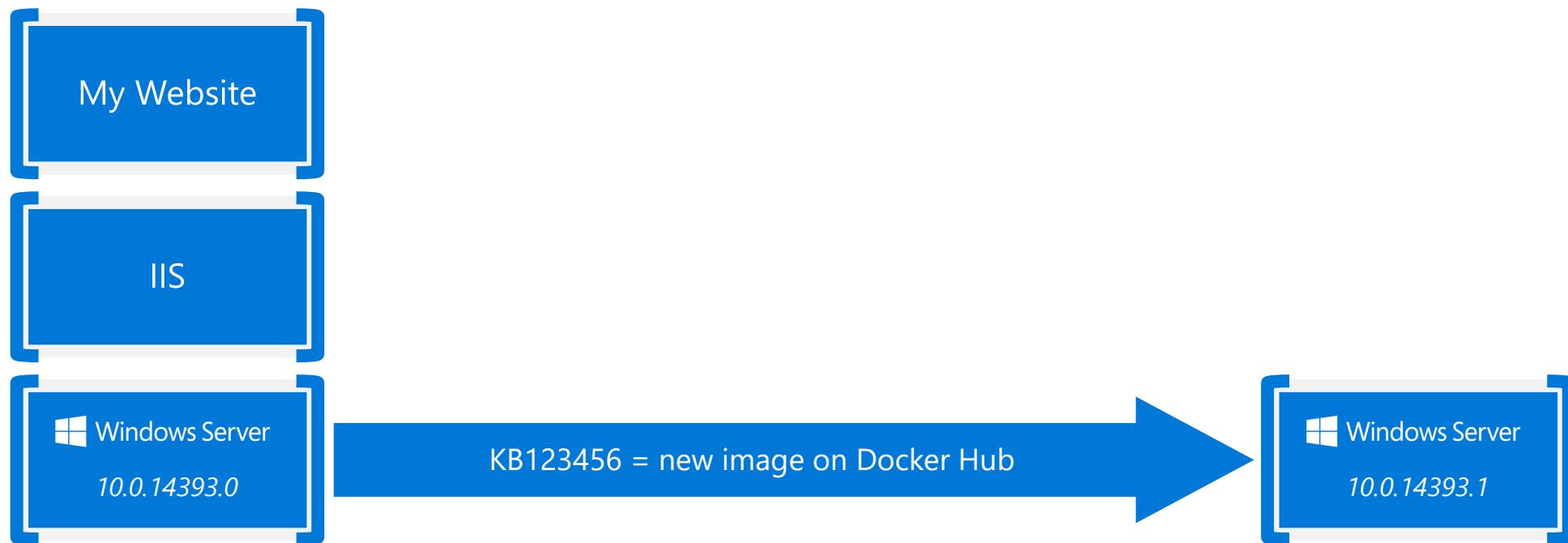


Patching and Updates

Update Container OS Image

Pull updated base image

Rebuild containers using dockerfiles

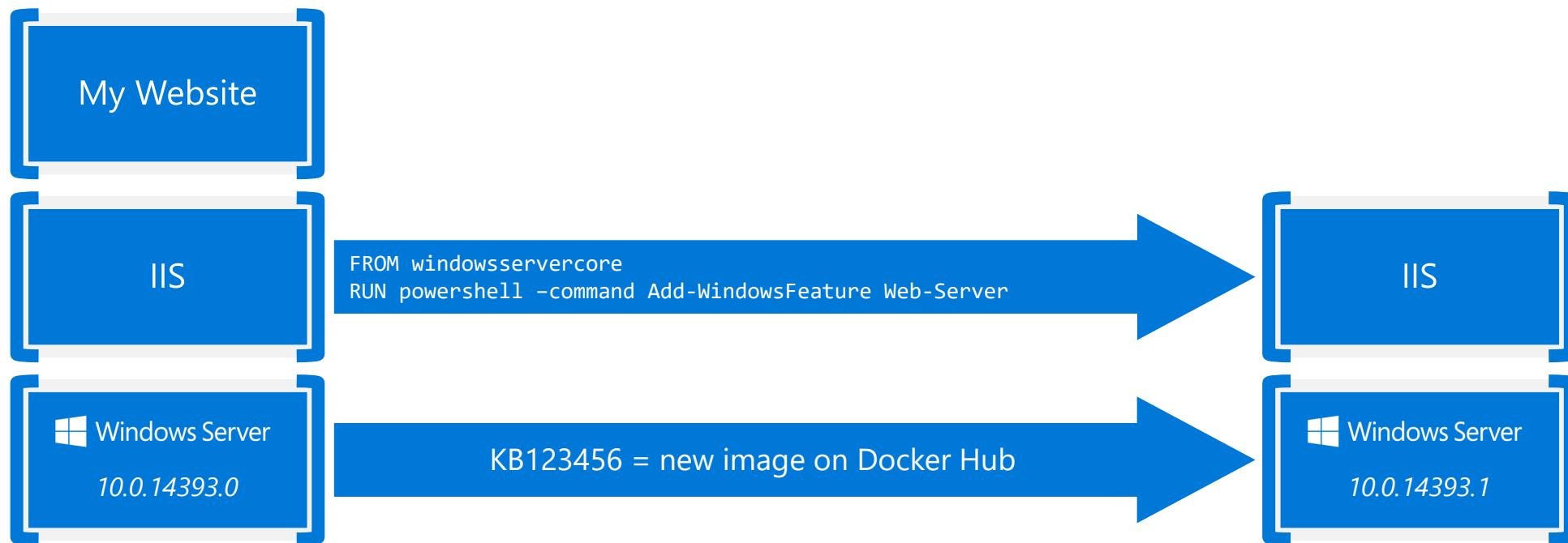


Patching and Updates

Update Container OS Image

Pull updated base image

Rebuild containers using dockerfiles

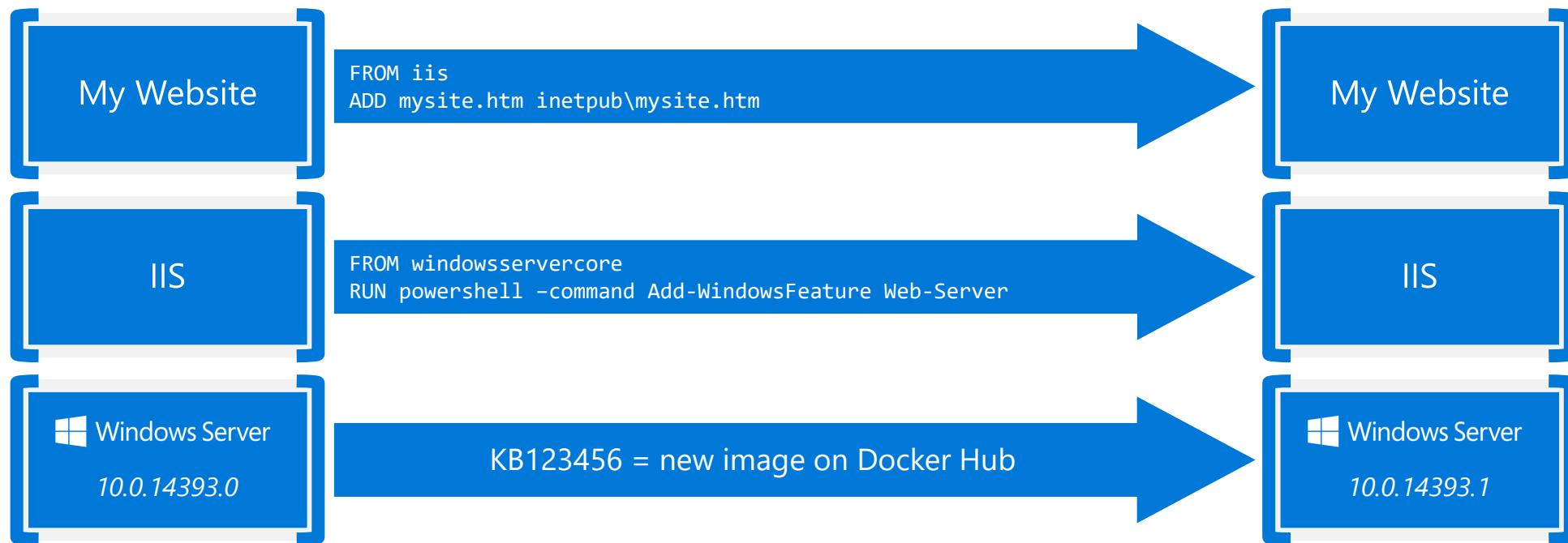


Patching and Updates

Update Container OS Image

Pull updated base image

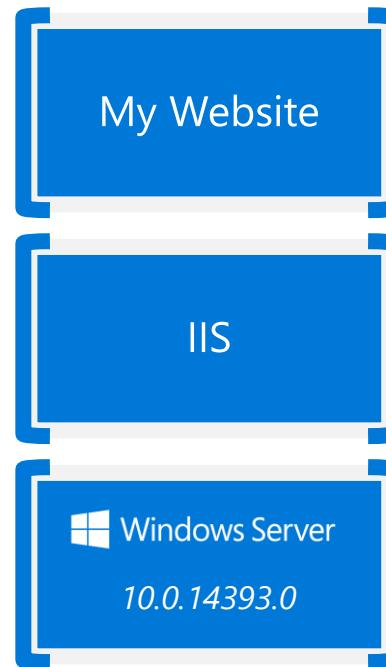
Rebuild containers using dockerfiles



Patching and Updates

Update as a new layer

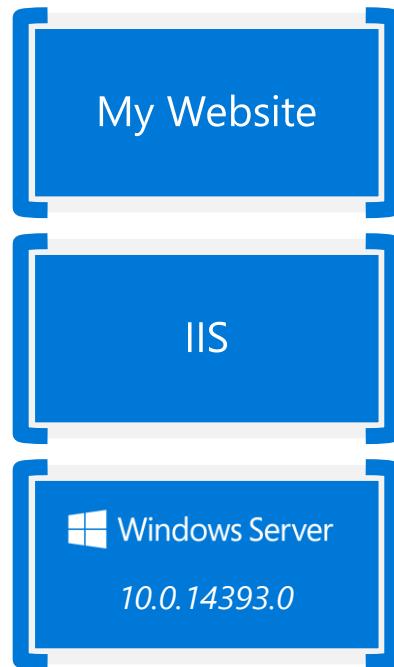
Download update in container (ala run Windows Update in the container)
When container is stopped update is applied as a new layer



Patching and Updates

Update as a new layer

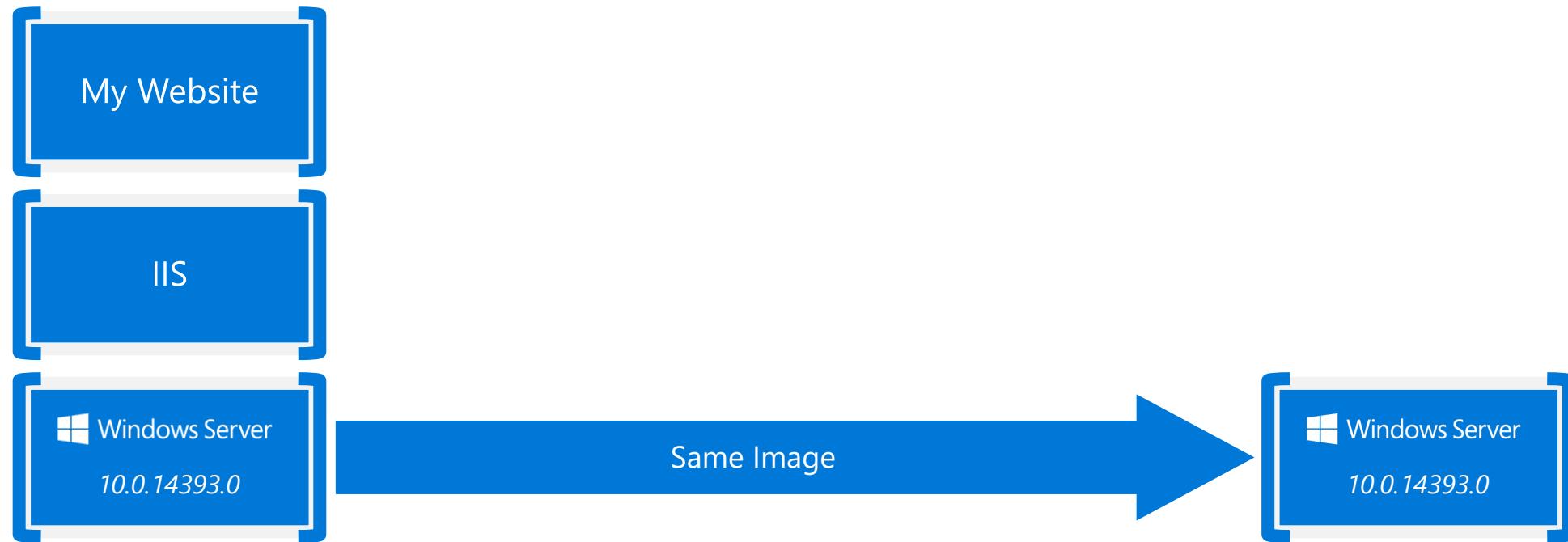
Download update in container (ala run Windows Update in the container)
When container is stopped update is applied as a new layer



Patching and Updates

Update as a new layer

Download update in container (ala run Windows Update in the container)
When container is stopped update is applied as a new layer



Patching and Updates

Update as a new layer

Download update in container (ala run Windows Update in the container)

When container is stopped update is applied as a new layer

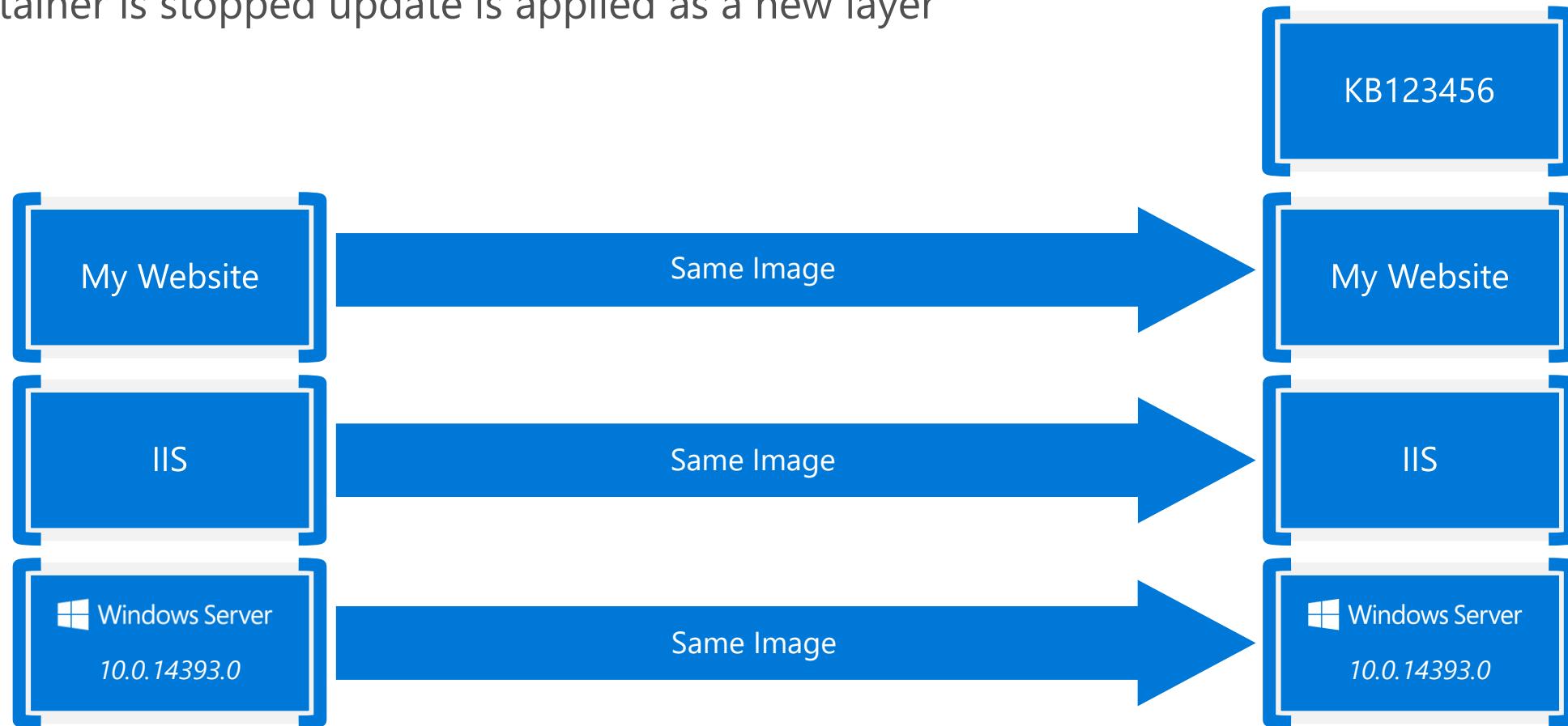


Patching and Updates

Update as a new layer

Download update in container (ala run Windows Update in the container)

When container is stopped update is applied as a new layer



Resource Controls

CPU

Percent of host CPU container can consume

Memory

Maximum memory container can consume

Disk

Maximum IO bandwidth on the system drive

Maximum IOPs limit on the system drive

Network

Platform support for egress caps

Networking

Fully Manageable with Docker

Network creation/enumeration
Service Discovery

Optimized for Microsoft Cloud Stack

Advanced network policy (ACLs, QoS) can be assigned per container endpoint
Load Balancing can be handled through the Microsoft Software Load Balancer (Coming Soon)

Area of Regular and Continuous Innovation

Docker tooling support (Compose) for networking (Limited Support at GA)
Integration with other Orchestrators (Kubernetes, Swarm, etc.)
Native Overlay network driver
Multiple networks (NAT and overlay) per host

Storage

Container Image

Not designed for persistent data

Not designed for secrets

“Volumes”

Enables storage persistence

Enables mapping of storage into containers

Read-Only or Read/Write

Multiple containers on the same host can access the same location

Plug-In Architecture

Network Storage

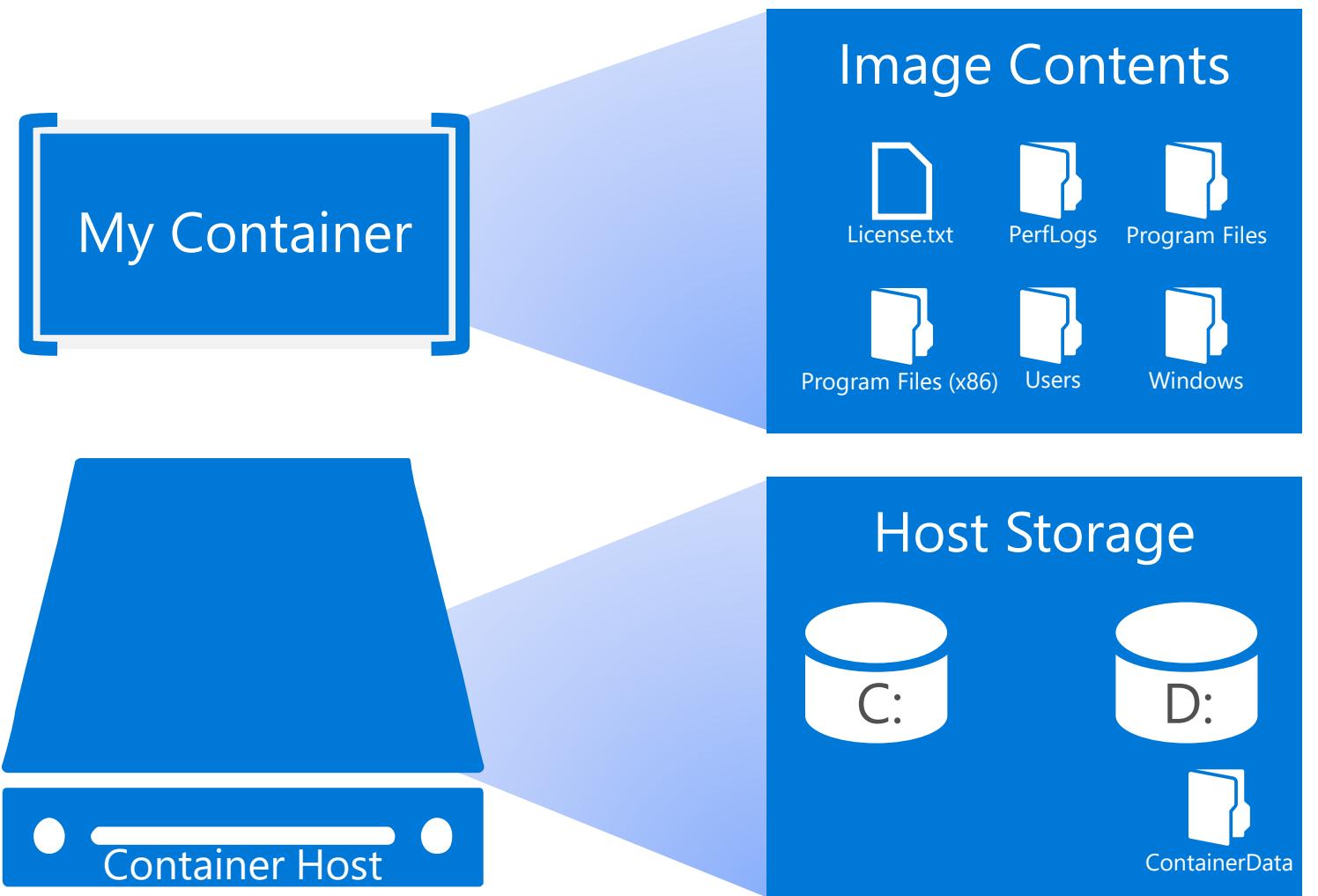
Containers access SMB shares

Accessed through the containers network

Volume Mapping



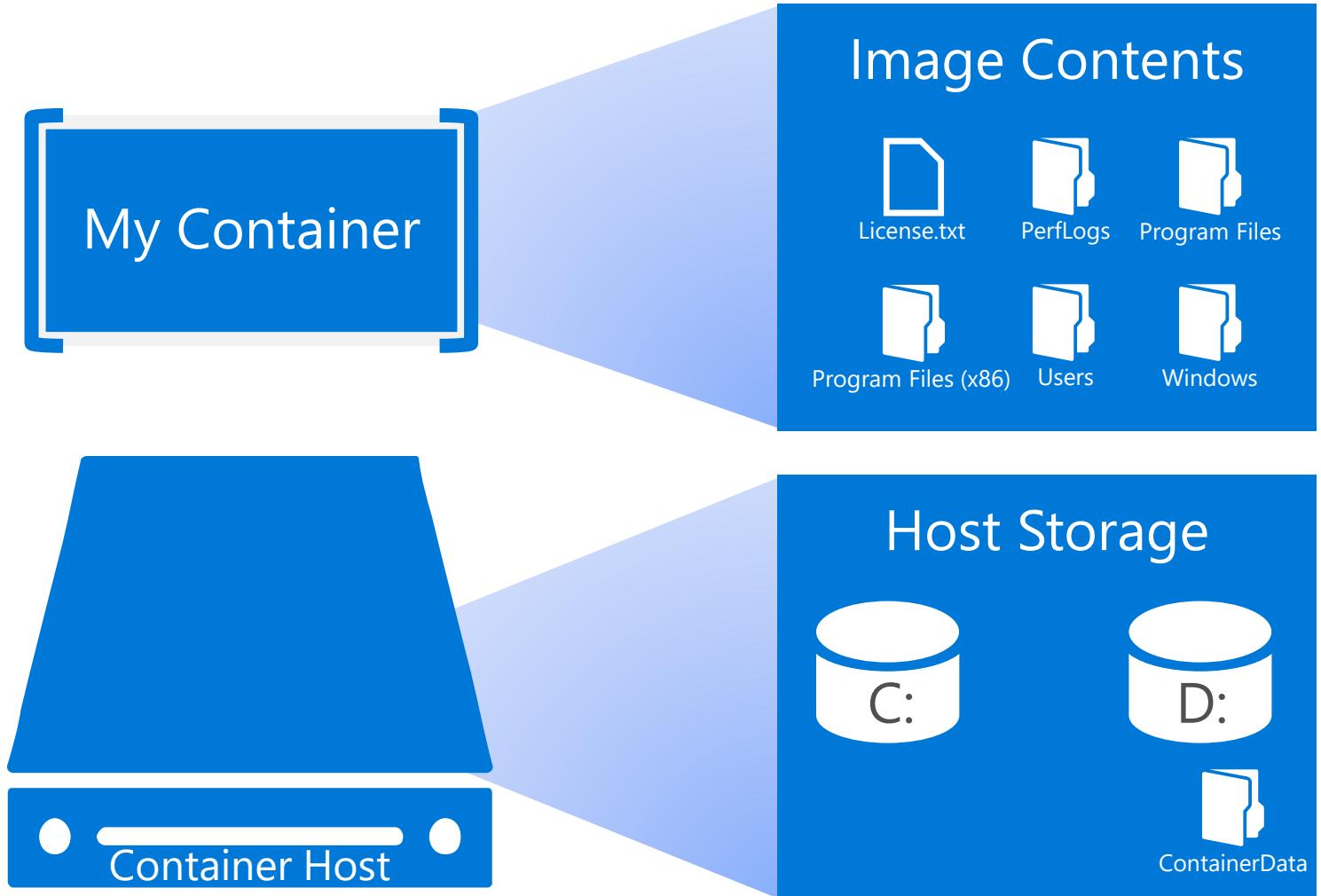
Volume Mapping



Storage Volume Mapping Example

Running a Container

```
docker run  
-v d:\ContainerData:c:\data  
mycontainer
```



Volume Mapping

Running a Container

```
docker run  
-v d:\ContainerData:c:\data  
mycontainer
```

Container View

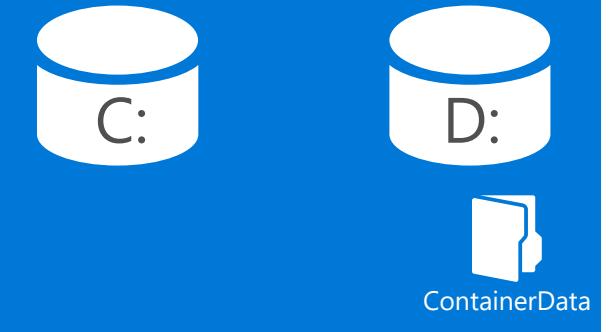


My Container

Image Contents

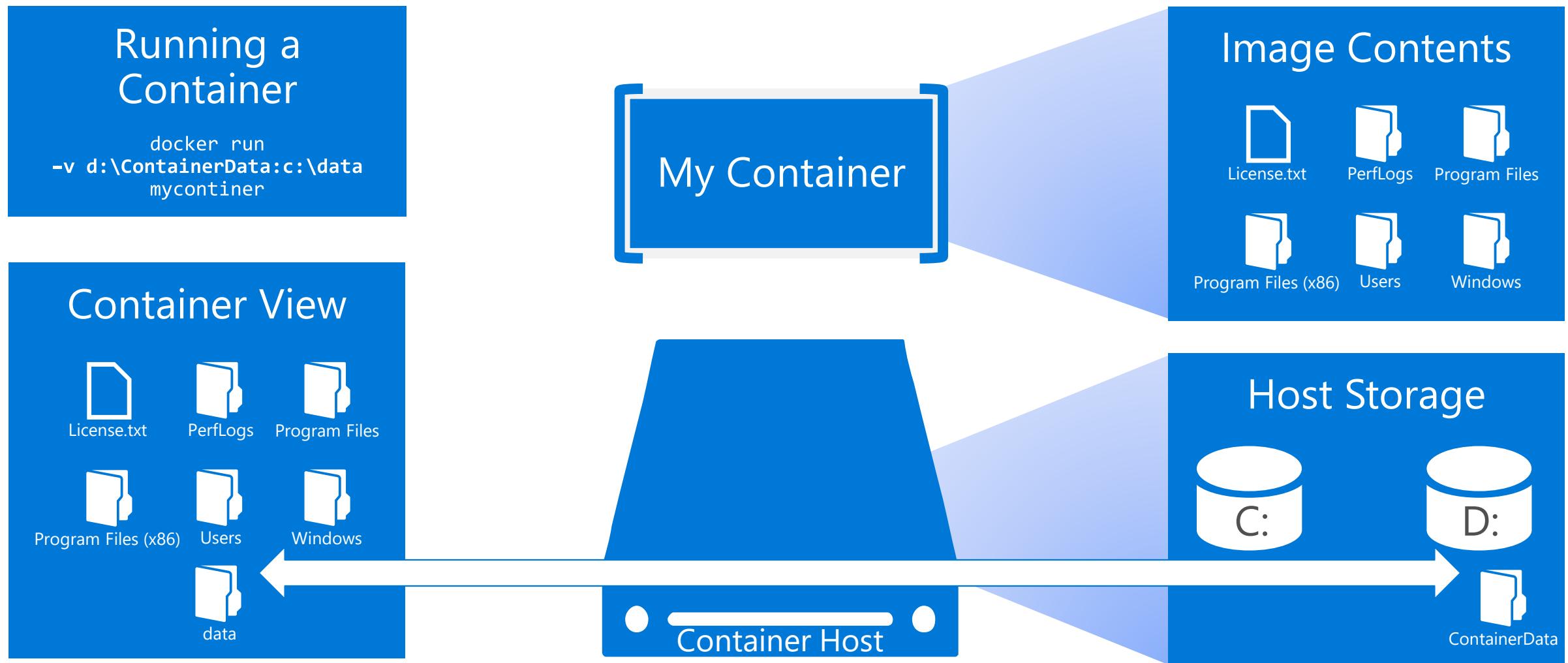


Host Storage



Container Host

Volume Mapping



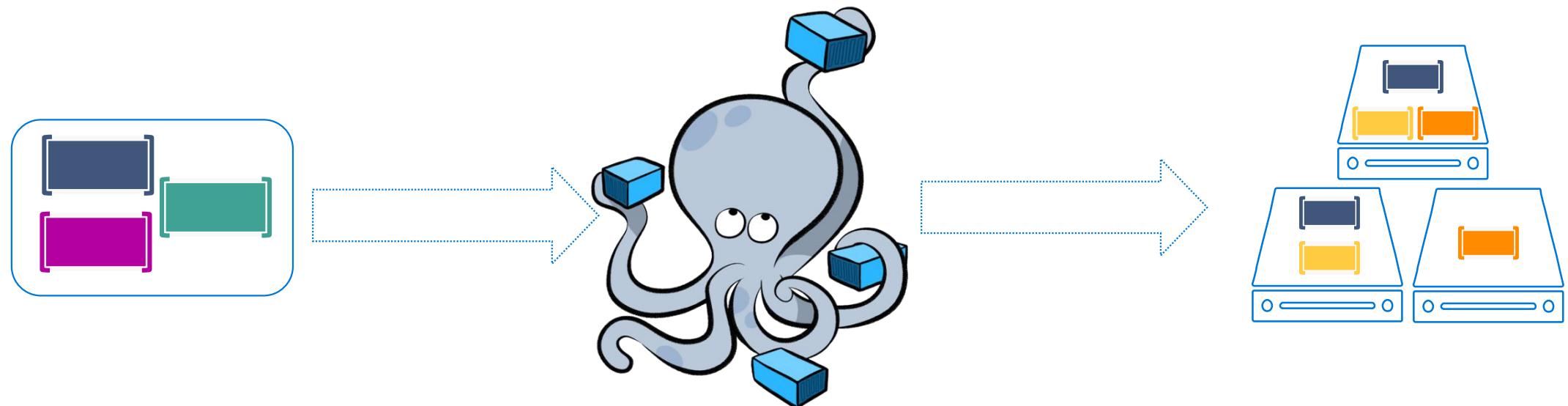
Automation and Management

Composition and Orchestration

Applications typically comprised of multiple containers

Containers typically hosted across a cluster of nodes

Orchestration tooling automates this



Composition and Orchestration

Docker Compose

Define application as separate containers

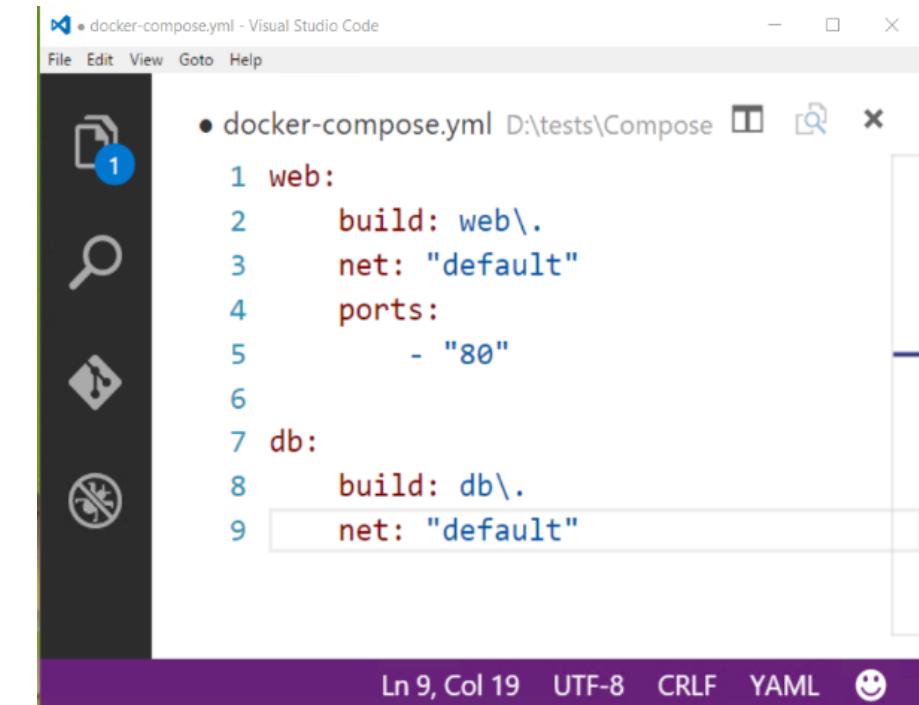
Manage different containers as a unit

Scale parts of application as needed

Docker Swarm

Aggregate container hosts

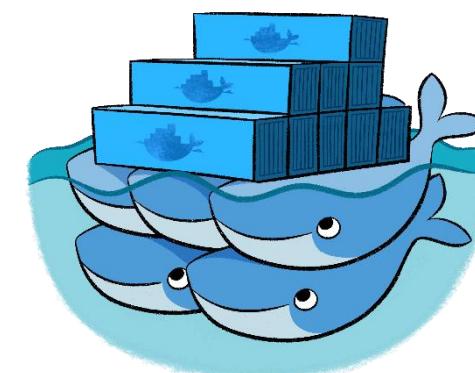
Supports tagging, affinity/anti-affinity



A screenshot of the Visual Studio Code interface showing a file named "docker-compose.yml". The file contains YAML configuration for a Docker Compose application. The code is as follows:

```
• docker-compose.yml D:\tests\Compose
File Edit View Goto Help
1 web:
2   build: web\.
3   net: "default"
4   ports:
5     - "80"
6
7 db:
8   build: db\.
9   net: "default"
```

The status bar at the bottom shows "Ln 9, Col 19" and "UTF-8 CRLF YAML".



Composition and Orchestration

Azure Service Fabric

Microservice and orchestration platform

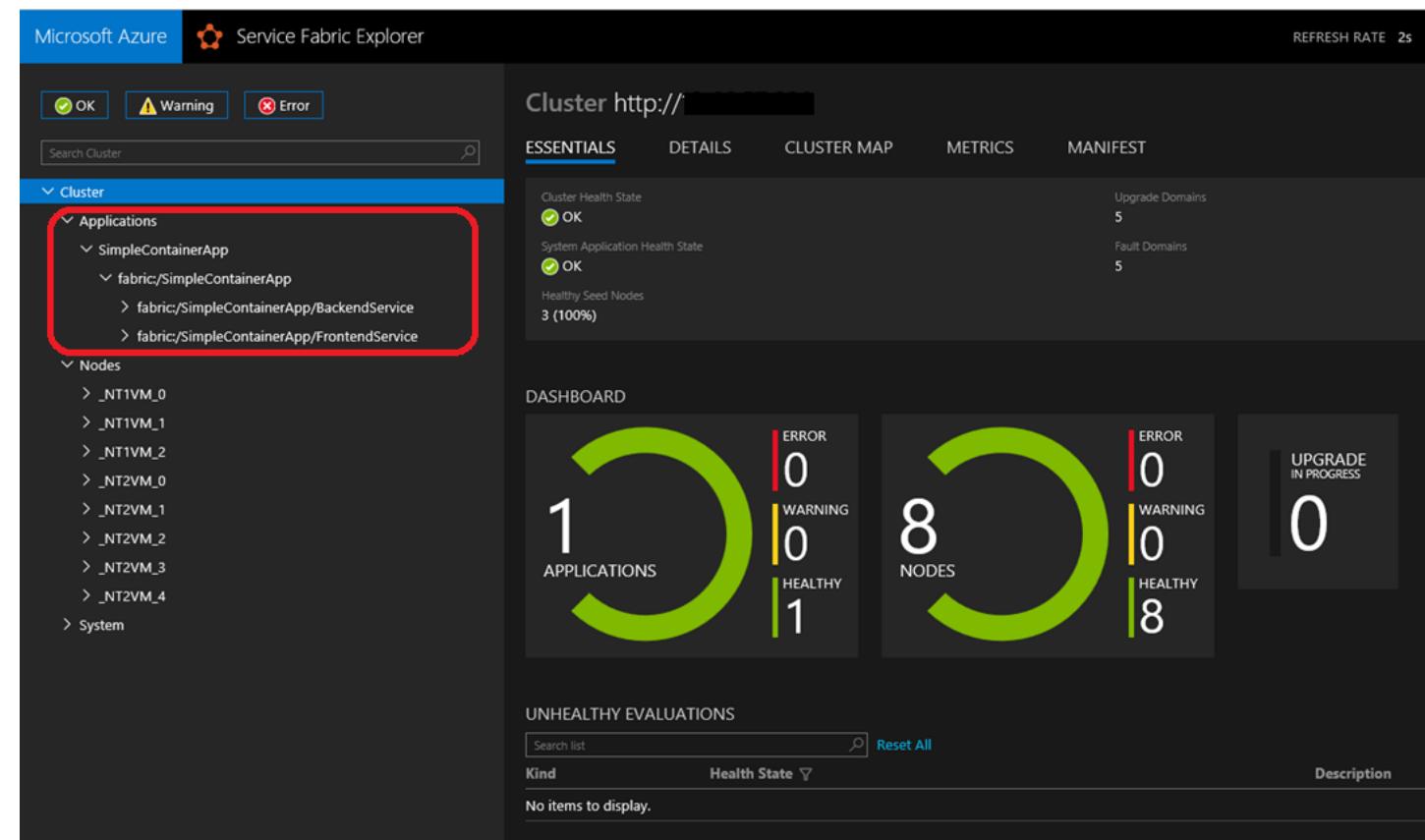
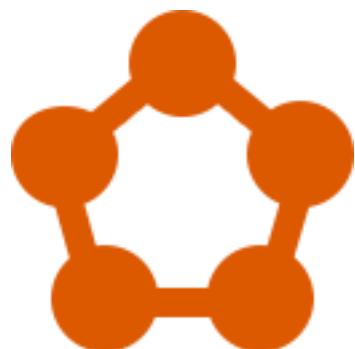
Build applications as containers **and/or** microservices

Available on Windows & Linux

Built-in cross-container communication

Web based management UI

Available On-Prem, Azure or other Clouds



The screenshot shows the Service Fabric Explorer interface for managing a cluster. The left pane displays the cluster structure under the 'Cluster' node, with the 'Applications' section expanded. A red box highlights the 'SimpleContainerApp' entry, which further expands to show its services: 'fabric:/SimpleContainerApp/BackendService' and 'fabric:/SimpleContainerApp/FrontendService'. The right pane provides a detailed view of the cluster's health across various metrics:

- ESSENTIALS:**
 - Cluster Health State: OK
 - System Application Health State: OK
 - Healthy Seed Nodes: 3 (100%)
- DASHBOARD:**
 - 1 APPLICATIONS (Green circle)
 - 8 NODES (Green circle)
 - 0 ERROR, 0 WARNING, 1 HEALTHY (Metrics bar)
- UPGRADE IN PROGRESS:** 0
- REFRESH RATE:** 2s

At the bottom, there is a table for 'UNHEALTHY EVALUATIONS' which is currently empty.

Composition and Orchestration

Kubernetes

Open source project started by Google

Windows support being added through community partnership spear headed by Apprenda



kubernetes

default ▾

+ DEPLOY APP UPLOAD YAML

Workloads

Replication controllers

Name	Labels	Pods	Age	Images	⋮
✓ anotherapp	app: anotherapp	2 / 2	11 days	ubuntu	⋮
✓ myapp	app: myapp	2 / 2	11 days	ubuntu	⋮
✓ redis-master	name: redis-master	1 / 1	11 days	redis	⋮
✓ redis-slave	name: redis-slave	2 / 2	11 days	gcr.io/google...redisslave:v1	⋮

Pods

Name	Status	Restarts	Age	Cluster IP	CPU (cores)	Memory (bytes)	⋮
✓ another...p-kt27y	Running	296	11 days	172.17.0.11	-	-	⋮
✓ another...p-ykrq3	Running	295	11 days	172.17.0.6	-	-	⋮
✓ frontend-660va	Running	0	11 days	172.17.0.2	-	-	⋮
✓ frontend-ive34	Running	0	11 days	172.17.0.4	-	-	⋮
✓ frontend-qb2je	Running	0	11 days	172.17.0.3	-	-	⋮
✓ k8s-etcd...7.0.0.1	Running	0	13 days	127.0.0.1	-	-	⋮

Composition and Orchestration

Mesos/Mesosphere + Marathon

Aggregates container hosts

Web based UI

Service Launch and Discovery



New Application

ID	simpleweb
CPUs	0.1
Memory (MiB)	256
Disk Space (MiB)	0
Instances	1

Command
May be left blank if a container image is supplied

Docker container settings

Image	yeasy/simple-web	Network	Host
-------	------------------	---------	------

Privileges

Extend runtime privileges to this container

Select to give this container access to all devices on the host

Port Mappings

Use the ports field to assign ports in host network mode.

Parameters

Key	Value	⊖	⊕
-----	-------	---	---

Volumes

Container Path	Host Path	Mode	Select
----------------	-----------	------	--------

Environment variables

Labels

Health checks

Optional settings

Executor

Executor must be the string '//cmd', a string containing only single slashes ('/'), or blank.

Ports

80

Comma-separated list of numbers. 0's (zeros) assign random ports. (Default: one random port)

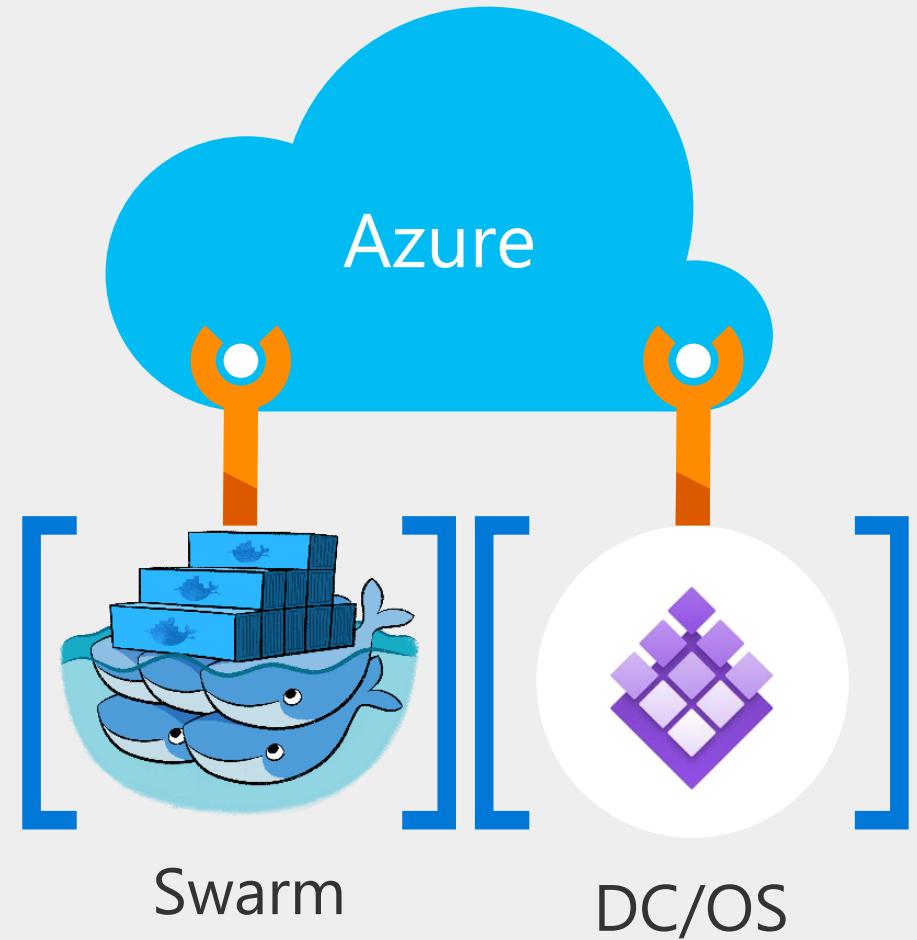
Azure Container Service

Standard Docker tooling
and API support

Streamlined provisioning
of Docker Swarm and DCOS

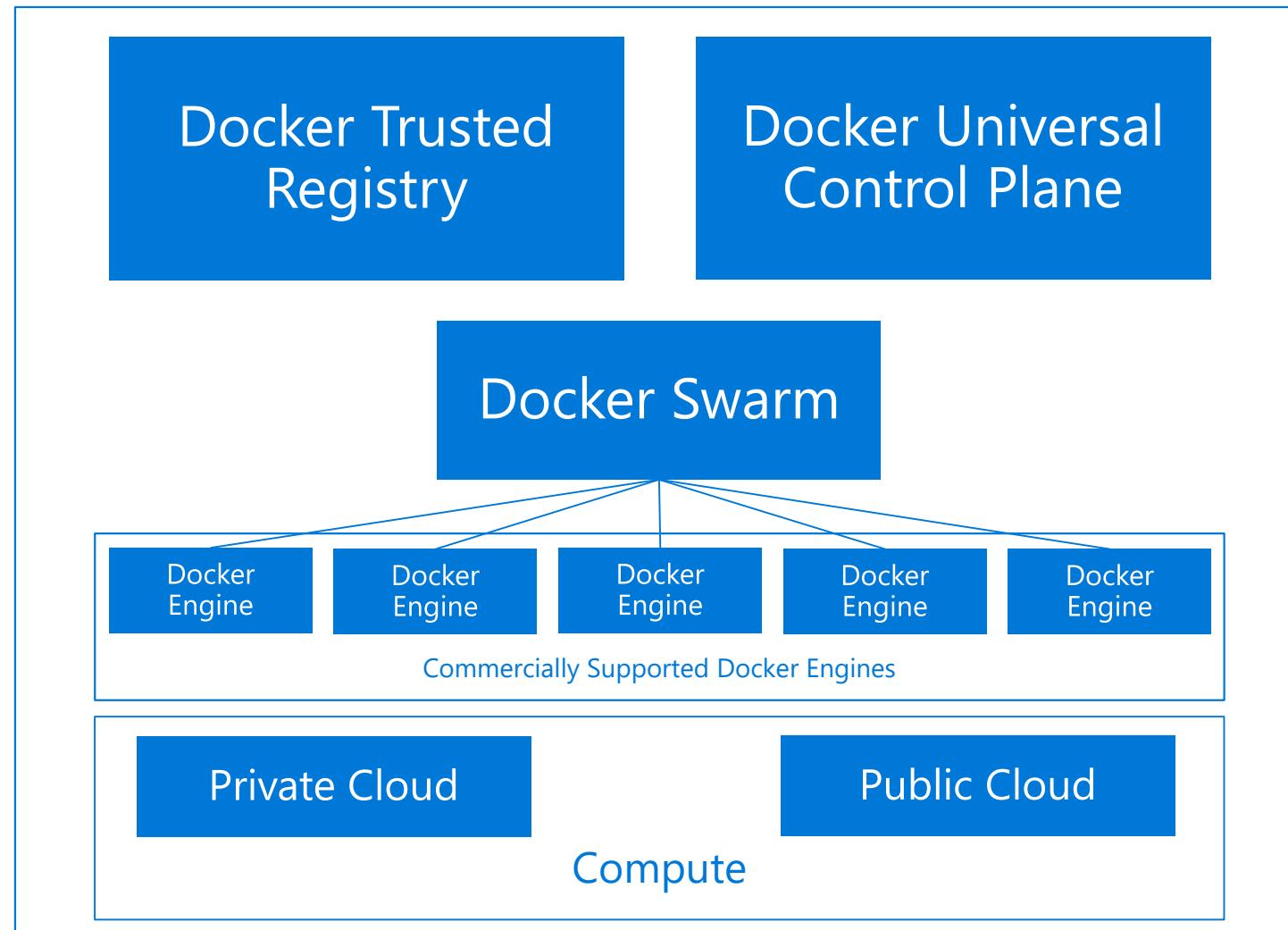
Linux and Windows Server
containers

Azure and Azure Stack



Management and Monitoring Tools

Docker Datacenter



Management and Monitoring Tools

Docker Datacenter

The screenshot shows the Docker Universal Control Plane (UCP) web interface. The left sidebar contains navigation links for Dashboard, Applications, Containers, Nodes, Volumes, Networks, Images, UCP Admin, Users & Teams (which is currently selected), and Settings. The main content area is titled "Dashboard / Users & Teams" and displays the "All Users" section. It includes a "Create User" button, a search bar, and a table listing five users:

USERNAME	FIRST NAME	LAST NAME	ROLE	OPTIONS
admin	Orca	Admin	Admin	Edit Delete
annie.jackson	Annie	Jackson	Admin	Edit Delete
dave.lauper	Dave	Lauper	Admin (You)	Edit Delete
dave.newport	Dave	Newport	Admin	Edit Delete
marie.carter	Marie	Carter	Admin	Edit Delete

At the bottom of the page, the footer reads "Universal Control Plane 1.1.0-dev (1fb5cef) | API: 1.22".

Microsoft Operations Management Suite

Monitoring Solution for Docker

Syslog events

Performance metrics

Container data

Overview > Containers

INFORMATION

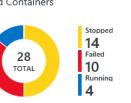
Container Solution
[More info](#)

Solution Overview
The container solution helps you see container usage and diagnose failures across your public and private cloud environments.

Supported Platform
This is only supported on Linux OS. For more detail on the supported Linux OS versions and Docker versions, please to [Github](#).

[See all...](#)

CONTAINER EVENTS

Failed Containers

28 TOTAL
14 Stopped
10 Failed
4 Running

Errors
Errors in the past 24 hours
2

CONTAINERS STATUS

1 Total Container Hosts	4 Total Running Containers
-------------------------	----------------------------

IMAGE **FAILED CONTAINERS** **COMPUTER** **ERROR COUNT** **CONTAINER HOST** **RUNNING CONTAINERS**

centos	8	redhatcontainer	2	redhatcontainer	4
ubuntu	2				

CONTAINER IMAGES INVENTORY

5 Total Images **4 Image Type Count**

IMAGES	COUNT
ubuntu	2
centos	1
hello-world	1
oms	1

CPU PERFORMANCE

Container Hosts CPU usage



AvgCPUPercent

6:00 PM 12:00 AM 6:00 AM 12:00 PM

MEMORY PERFORMANCE

Container Hosts Memory usage



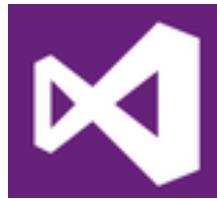
AvgUsedMemory

6:00 PM 12:00 AM 6:00 AM 12:00 PM

NOTABLE QUERIES

- See all the running versions of container <nginx> by tag
`Type=ContainerImageInventory Image=nginx | measure count()`
- Count the number of pull commands by computer:
`Type=ContainerServiceLog Command=pull | measure count() by...`
- See all containers that are running the <drupal> image
`Type=ContainerImageInventory Running>0 Image = drupal`
- See the CPU usage of all containers:
`Type = Perf ObjectName= "Container" CounterName= "% Proces...`
- See the Memory usage of a container <ContosoContainer>
`Type=Perf ObjectName= "Container" CounterName= "Memory U...`
- See the Network usage of a container <ContosoContainer>
`Type=Perf ObjectName= "Container" CounterName= "Network R...`
- See storage performance for container <ContosoContainer>
`Type=Perf ObjectName= "Container" CounterName= "Disk Write...`
- Get all logs for container <ContosoContainer>
`Type=ContainerLog Name=ContosoContainer | select LogEntry`
- See all the commands in past 24 hours.
`Type=ContainerServiceLog TimeGenerated > NOW-24HOURS`

Development Tools



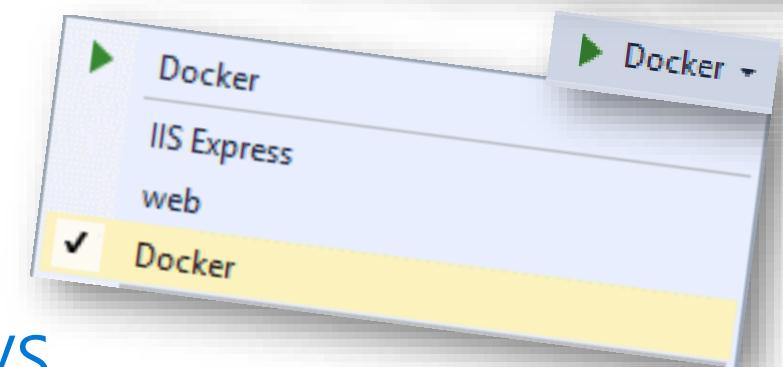
Visual Studio Docker Tools

- Run, Debug, Test Web & Console apps in docker containers
 - Linux today, Windows Server & Nano Server coming soon
- F5 Debugging
- Edit & Refresh of code
- Scaffolds docker assets
 - Dockerfile, docker-compose.yml

```
Dockerfile
FROM microsoft/aspnetcore:1.0.1
ARG source=.
WORKDIR /app
EXPOSE 80
COPY $source .
ENTRYPOINT ["dotnet", "Web.dll"]
```

```
Dockerfile
FROM microsoft/dotnet:1.0.1-core
ARG source=.
WORKDIR /app
COPY $source .
ENTRYPOINT ["dotnet", "BatchJob.dll"]
```

The screenshot shows the Visual Studio Tools for Docker - Preview extension page on the Visual Studio Marketplace. The page includes the extension's logo, name, developer information (Microsoft Cloud Explorer), rating (4 stars from 14 reviews), version (0.40.0), download count (34,795), update frequency (9/22/2016), and social sharing links. A green 'Add to favorites' button is also present. The page also lists supported platforms: Azure, linux, virtual machines, Docker, Containers, ASP.NET Core.



aka.ms/DockerToolsForVS



EXPAND MENU FOR MORE TOPICS

Windows/
ServerActive
Directory

Exchange

Cloud/Virt

SharePoint

Security

HOME

NEWS

FEATURES

PAPERS

WEBCASTS

SOFTWARE TRIALS

ADVERTISE

EVENTS

NEWSLETTERS

FREE SUBSCRIPTION

NEWS



2



Tweet



Share



44



Share 0

Windows Server 2016 Getting Linux Containers

By Scott Bekker ■ 04/20/2017

Microsoft this week announced that a new type of Linux container that runs on Windows Server 2016 will be coming soon.

The move, which was unveiled at this week's DockerCon 2017 in Austin, Texas, would break down a fundamental wall in deployment scenarios for containers to date. For now, Linux containers can only run on Linux host operating systems and Windows containers can only run on Windows



Most Popular Articles

[OpenSSH Windows Nearing Completion](#)[Does Everyone Still Need Windows?](#)[Windows 10 Creators Update Tools and Documentation Released](#)[Microsoft Releases Windows 10 Creators Update and Security Patches](#)[A Brave New World: Top Features Coming to SQL Server 2017](#)

