Microsoft

Containers as Infrastructure

Getting Started with Windows Containers

Microsoft Services

Objectives

- Understand Windows Containers Architecture
- Working with Nano Server and Windows Server Base OS Images
- Learn about Windows Container Layering.
- Build and Run IIS Server, ASP.NET 4.5 Web Application ASP.NET Core Application and SQL Server 2016 Containers
- Learn about Tools such as Visual Studio Support for Docker
- Learn Features of Windows Containers
- Active Directory Service Accounts for Windows Containers

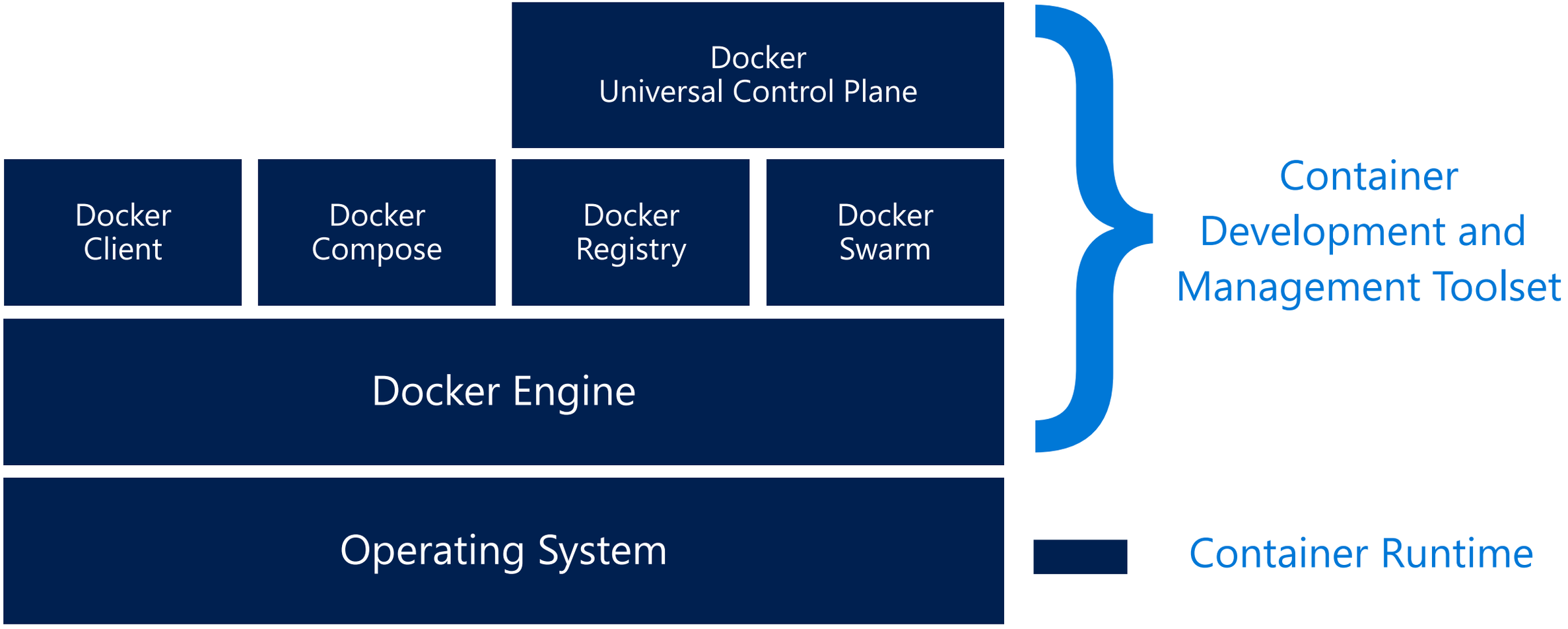
Microsoft

Getting Start with Windows Containers

Architecture in Windows

Microsoft Services

Architecture



Type of Windows Containers

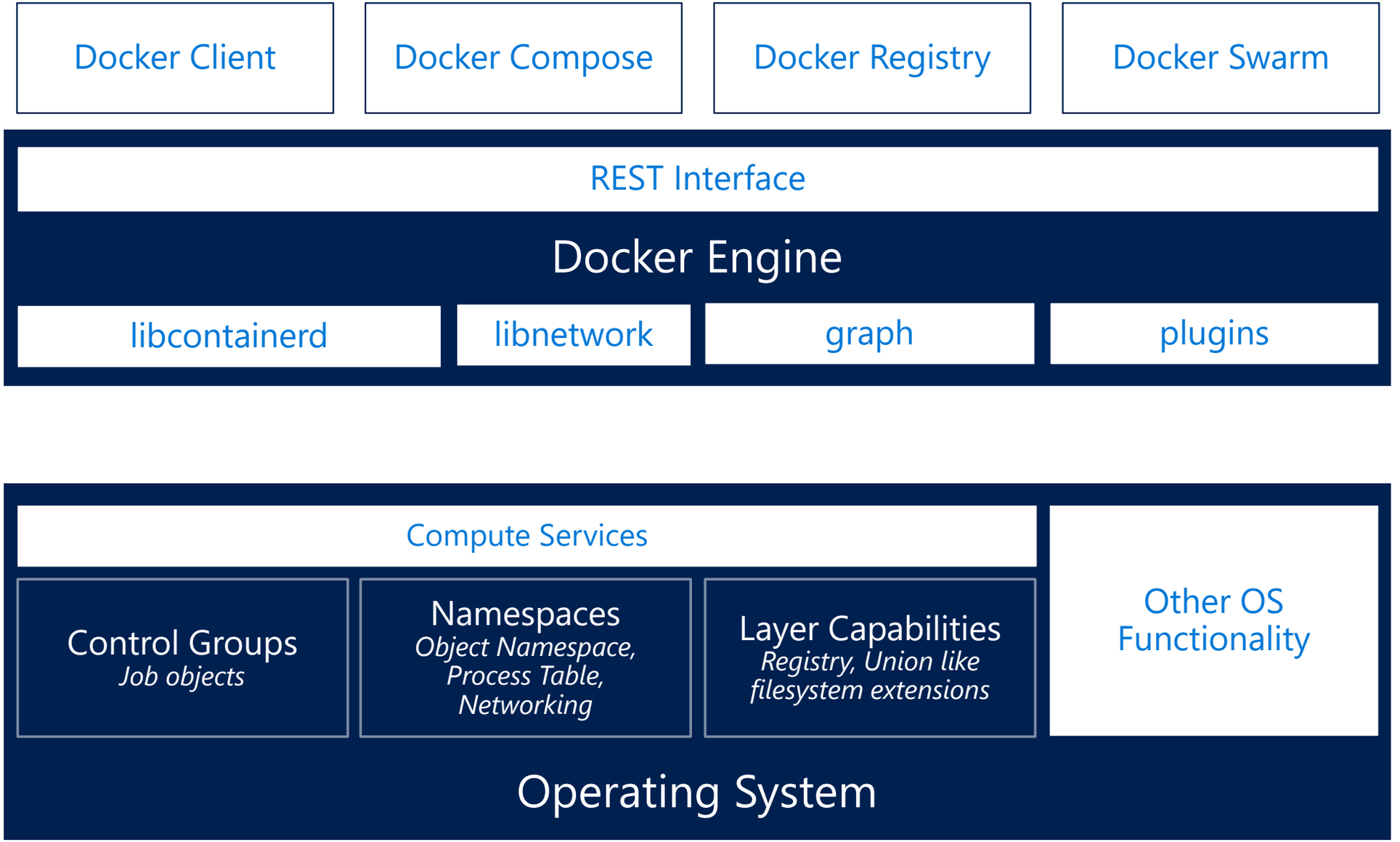
Windows Server containers

- Multiple container instances can run concurrently on a host, with isolation provided through namespace, resource control, and process isolation technologies.
- Windows Server containers share the same kernel with the host, as well as each other.

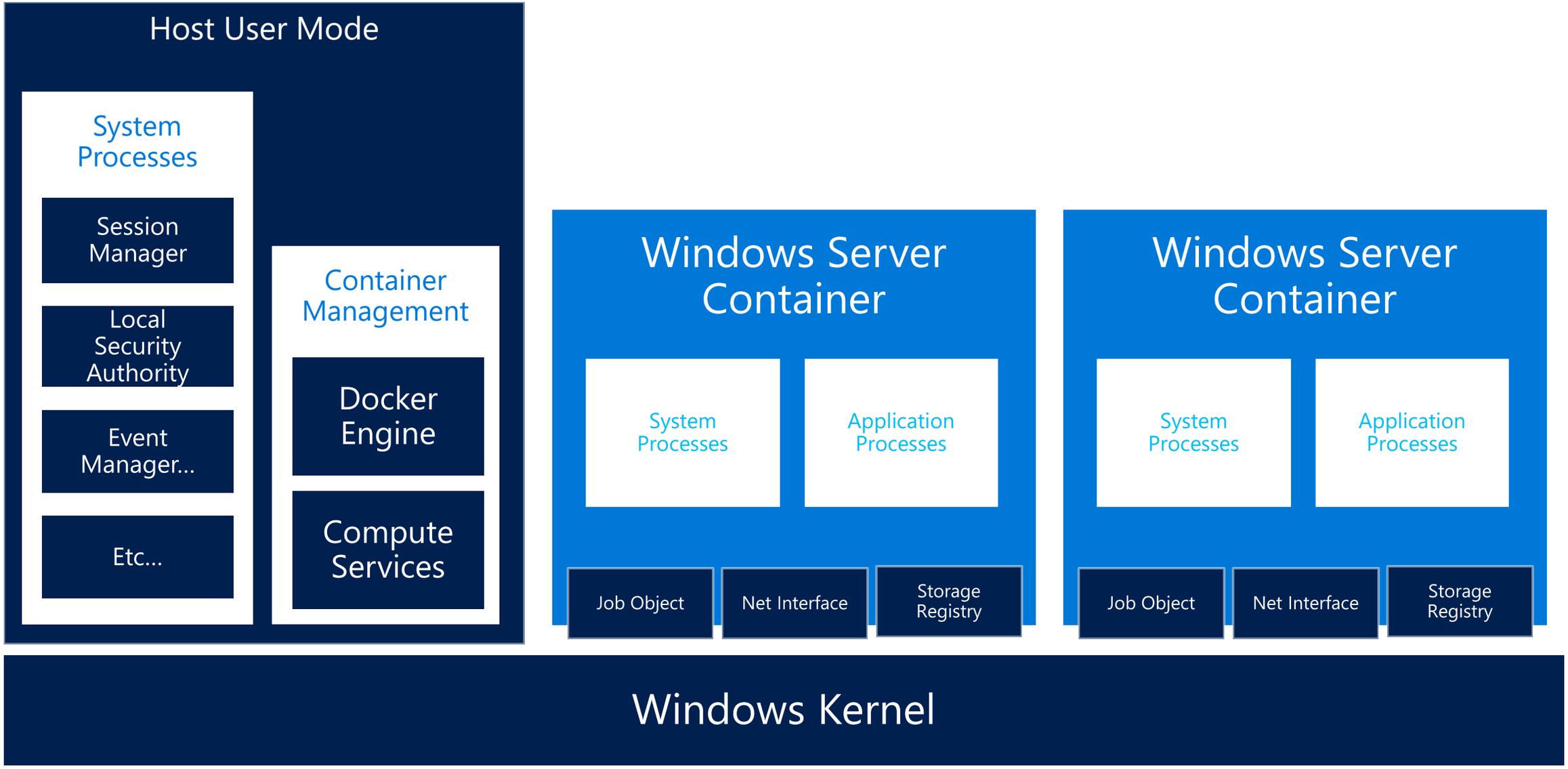
Hyper-V containers

- Multiple container instances can run concurrently on a host; however, each container runs inside of a special virtual machine.
- This provides kernel level isolation between each Hyper-V container and the container host.

Architecture in Windows

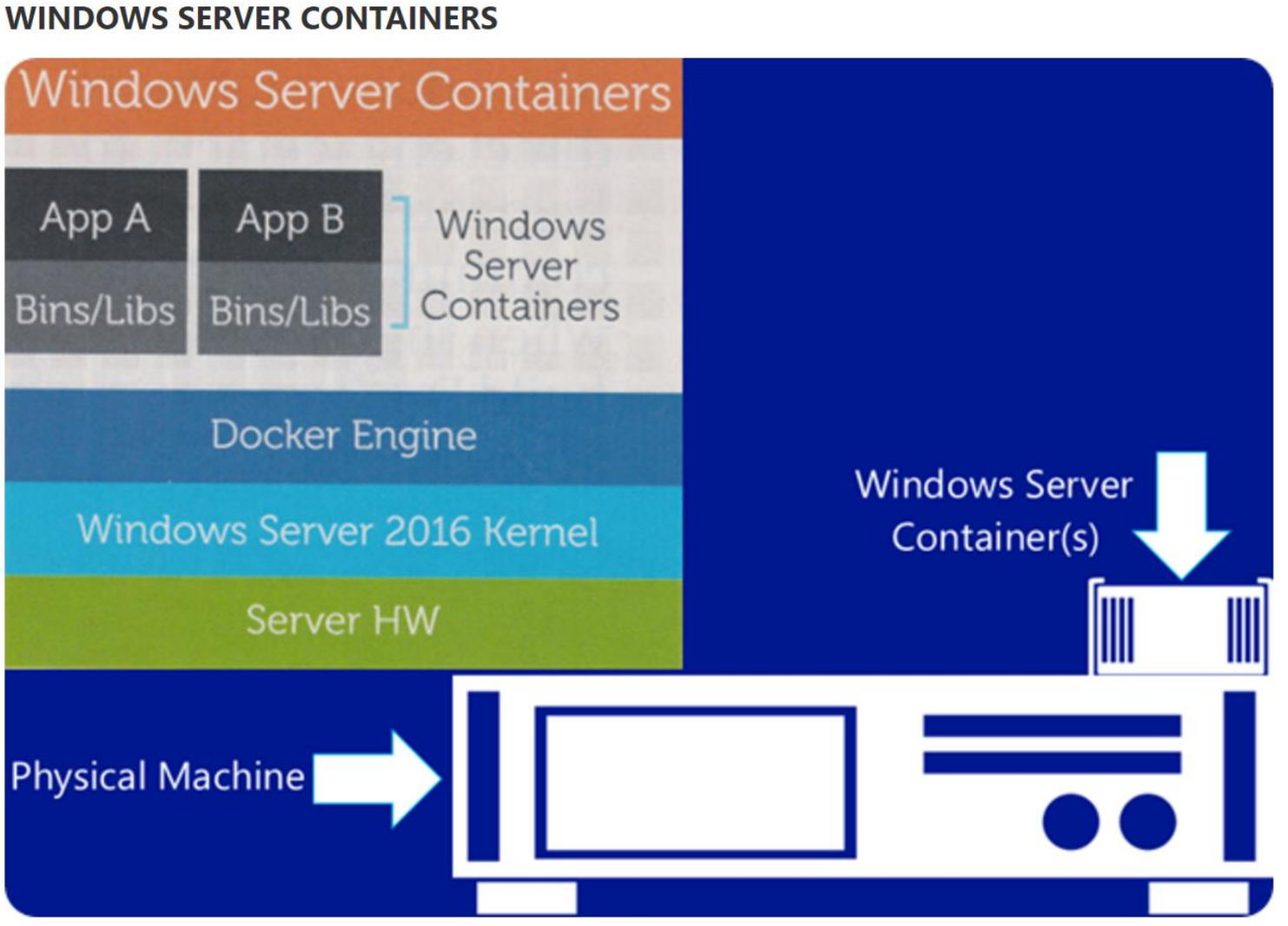


Windows Server Containers

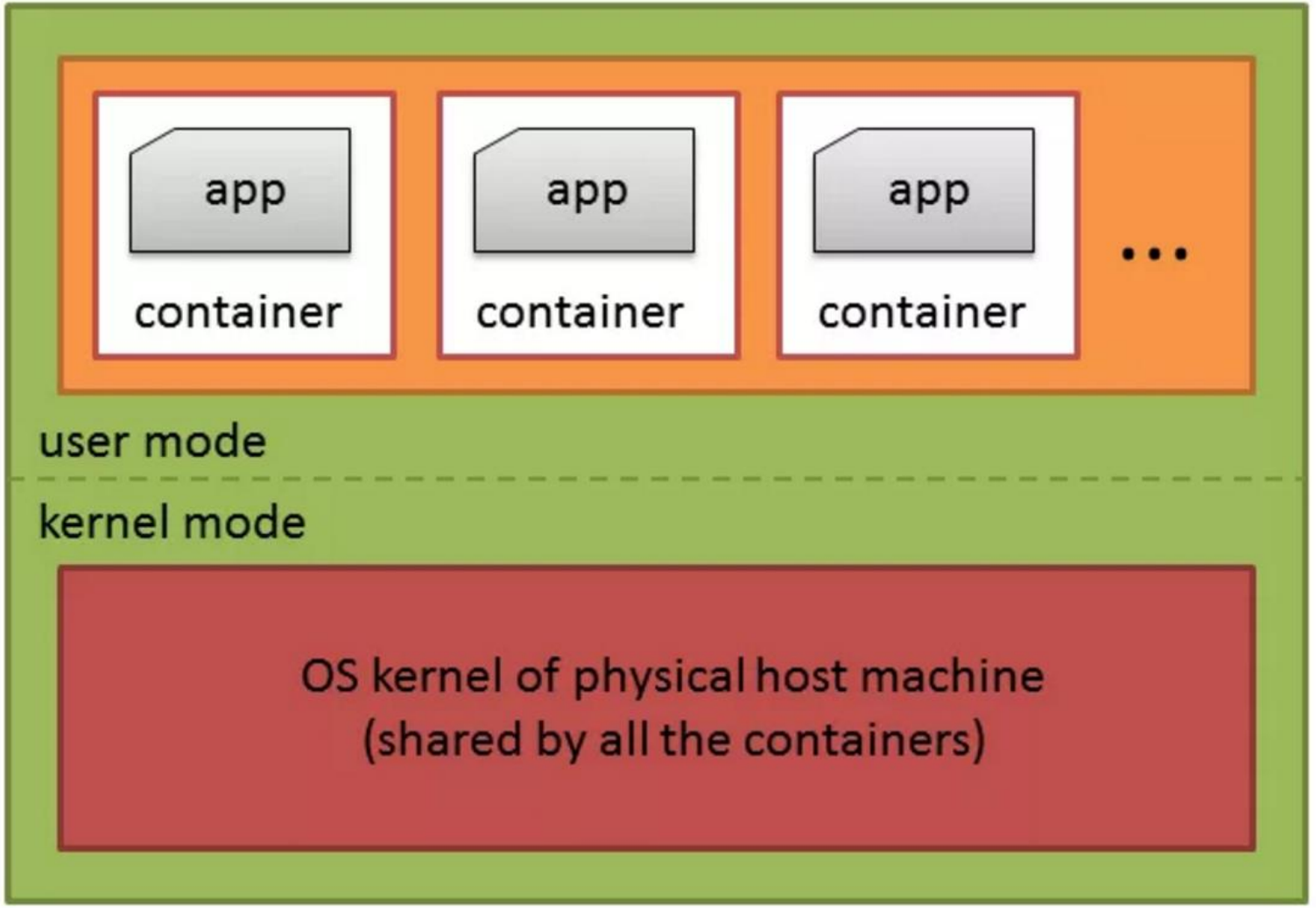


Windows Server Containers

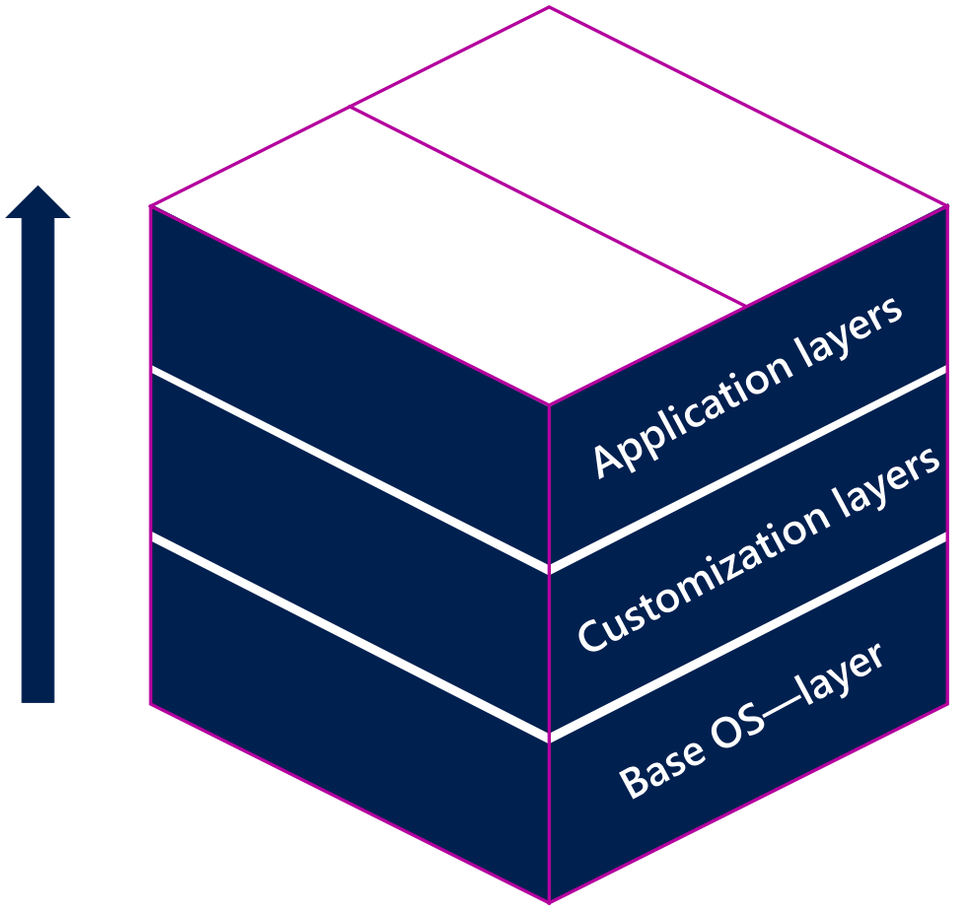
- Registry virtualization and network virtualization allows each application/container to have its own IP address;
- Provides a layer of isolation, so the container doesn't have access outside of its Sandbox
- Windows containers do not depend on Docker technology to work in Windows Server 2019



Windows Containers Shared Kernel Model



Anatomy of a Windows Container I



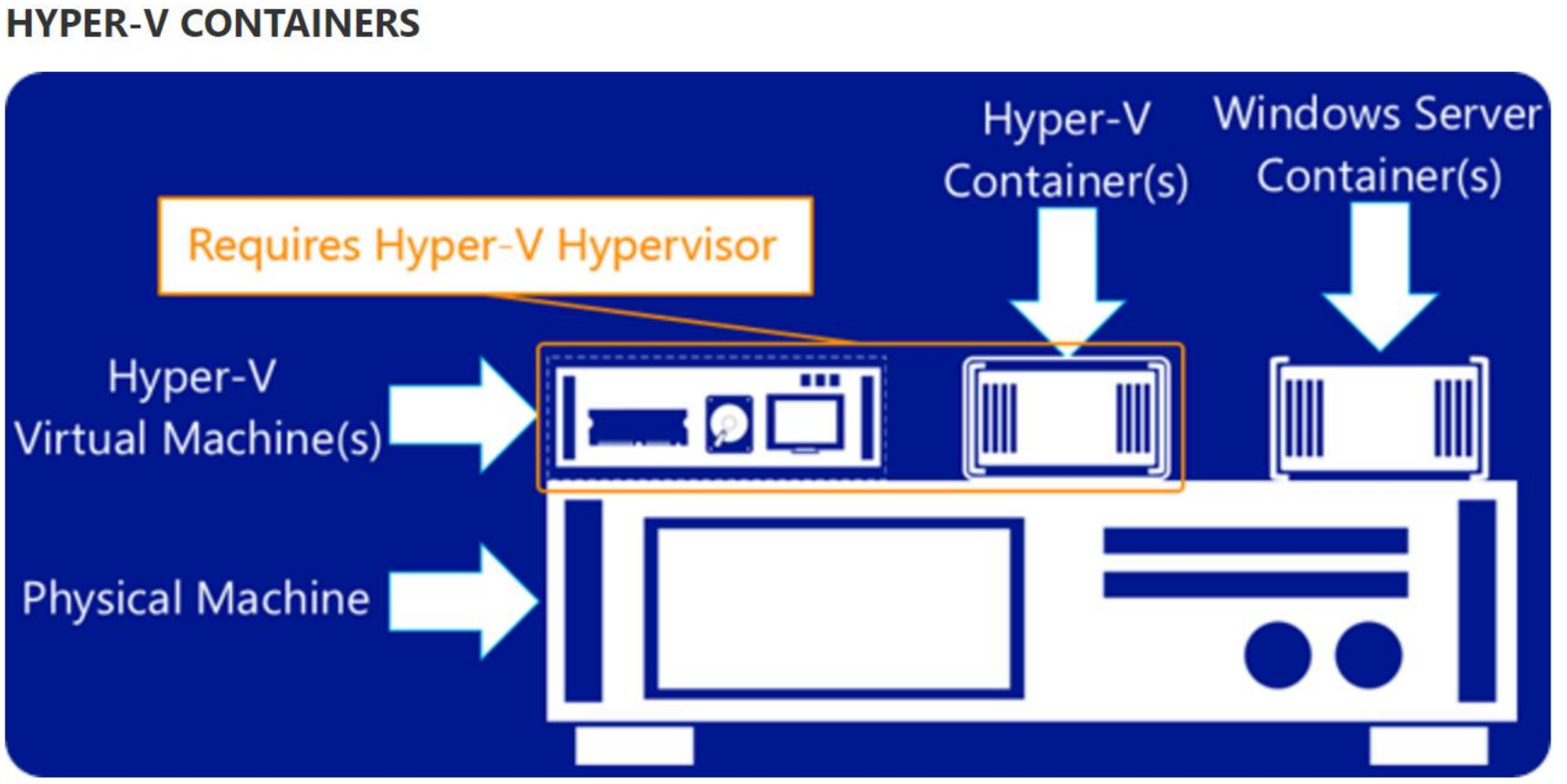
e.g. eShop

e.g. IIS, Powershell

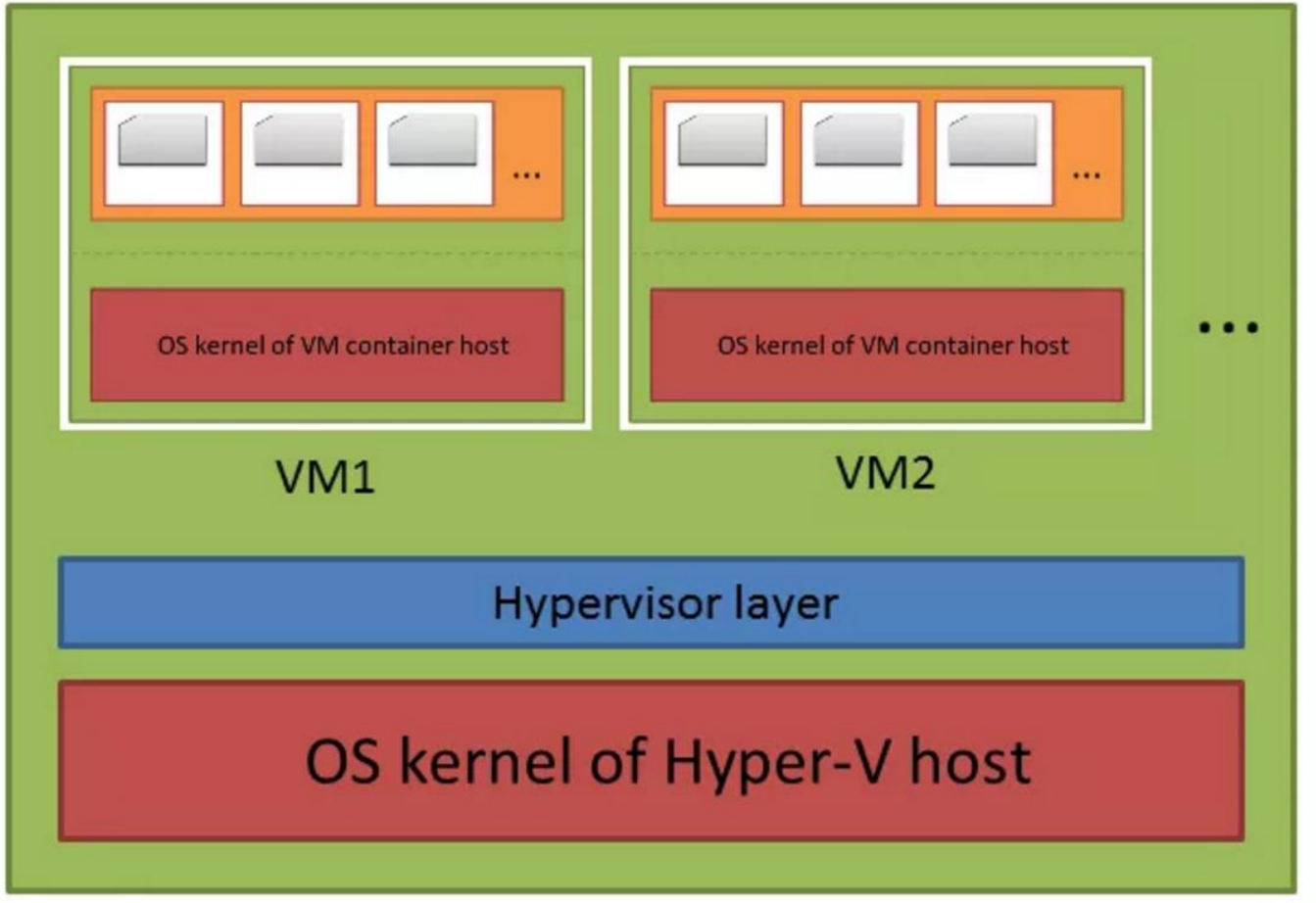
Windows Server Core, or Nano Server

Differences in Hyper-V Containers

- Offer both OS virtualization and machine virtualization
- Interchangeable with Windows Server Containers

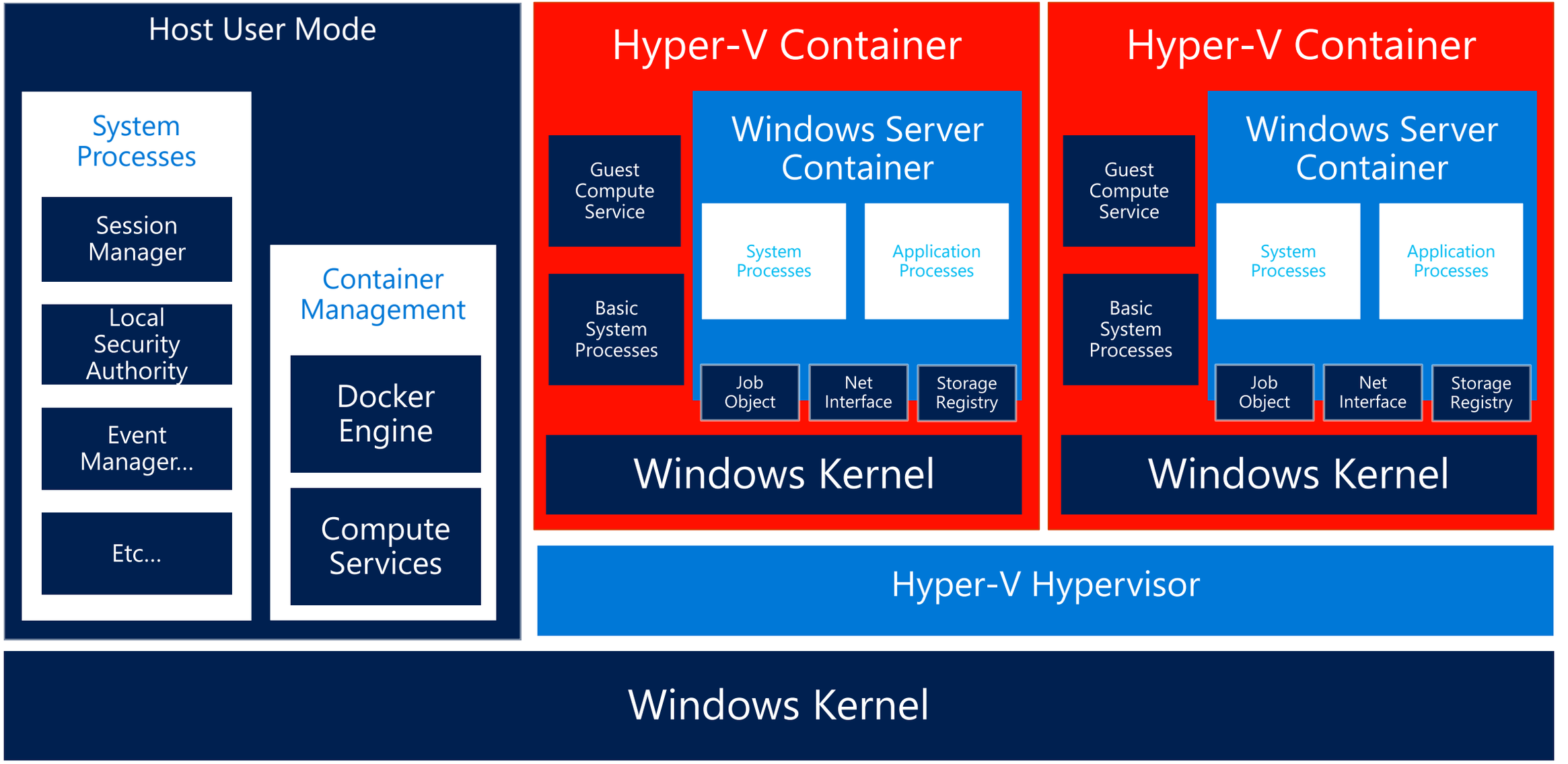


Hyper-V Containers Isolated OS Kernel Model

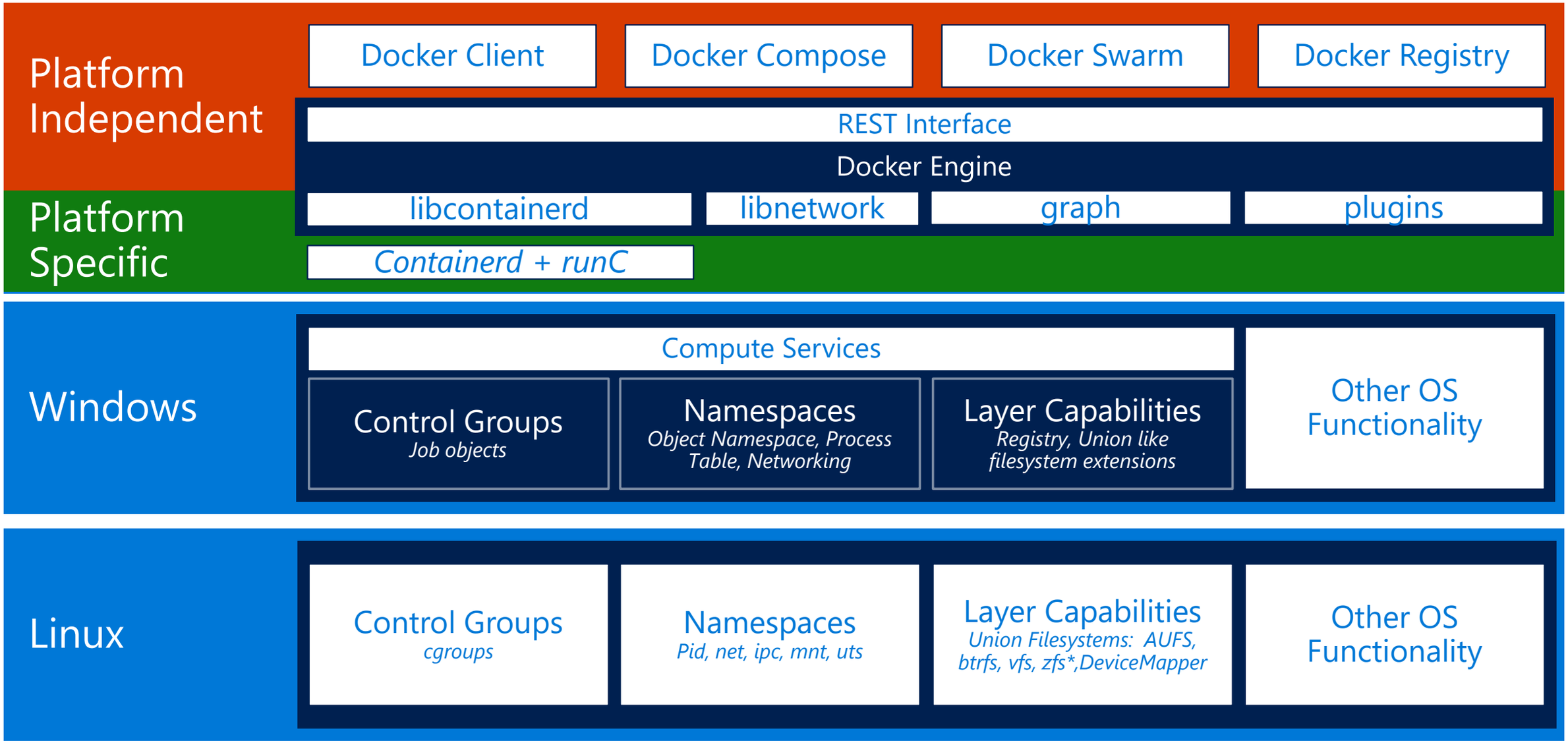


Hyper-V Containers

Hyper-V Containers

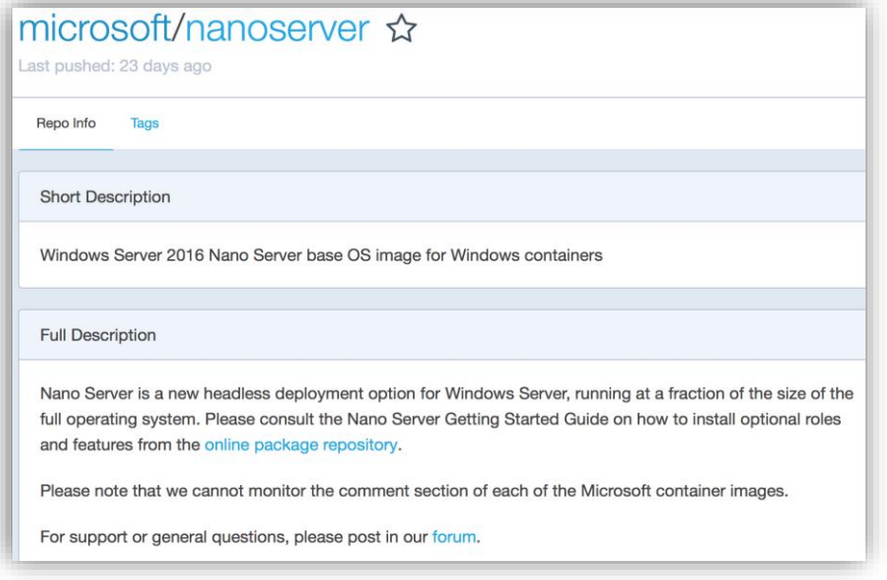


Comparing OS Architectures



Windows Container Base OS Images

- Base OS image is the first in potentially many image layers that make up a container, providing the OS environment
- Container OS Base Image is immutable
- Nano Server is a headless Windows Server that only support x64 applications
- Windows Server Core provides you with a minimal installation of Windows Server 2016 that supports installing only certain server roles



Containers: Over the last two years

Windows Server 2016

Initial launch of containers

Process and Hyper-V isolation

Docker EE Basic Included at no additional cost

Windows Server version 1709

Optimized container images for Nano Server and Server Core

Platform level support for Linux containers

Windows Subsystem for Linux

Networking enhancements for overlays and SDN

Windows Server version 1803

Server Core image Optimizations

Native command line tools – curl.exe, tar.exe and SSH

Enhancements to the Windows Subsystem for Linux

Networking enhancements for greater density and quicker endpoint creation

Improved network security with Calico

Open source storage plugins for Kubernetes

Platform functionality required for Kubernetes conformance

Windows Server 2019

Server Core image Optimizations

Improved compatibility for apps running in containers

Enhanced Group Managed Service Account support

Platform functionality for Kubernetes and Microsoft Service Fabric

Container Performance and density improvements

Platform and open source work on CNI networking plugins such as Calico and Flannel

Enhancements to the Windows Subsystem for Linux

Demonstration: Nano Server and Windows Server Core

Working with Nano Server Container

Working with Windows Server Core Container

Microsoft

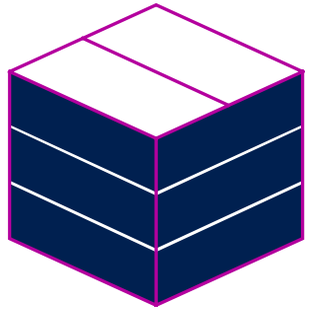
Getting Stated with Windows Containers

Container Management with Docker

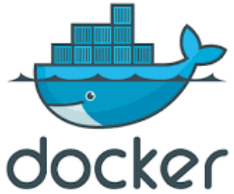
Microsoft Services

Fundamentals of Containers

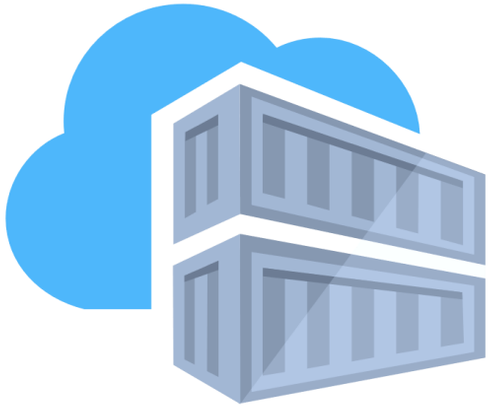
Images, Hosts, Repositories, Docker and Orchestrators



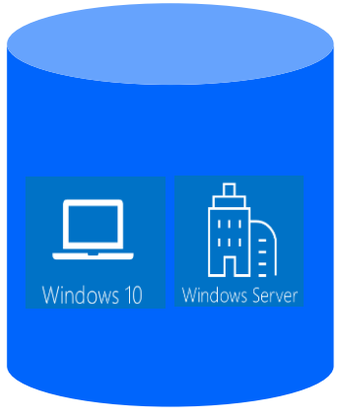
Containers are created from a **Container Image**



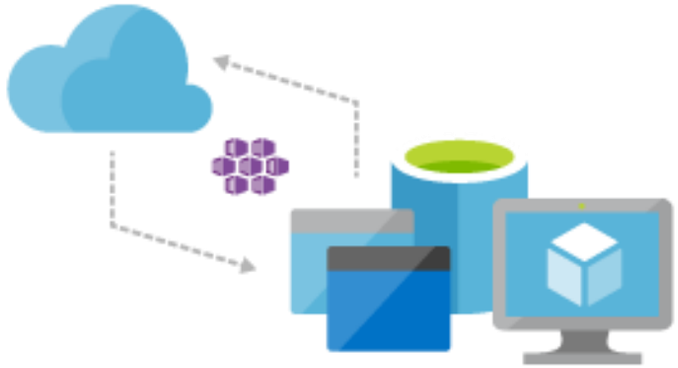
Containers are managed with **Docker**



Container Images are stored in a **Container Repository**



Containers are hosted on a **Container Host**



Containers are distributed and updated on Container Hosts with an **Orchestrator** (like Kubernetes or Docker Swarm)

Container Management

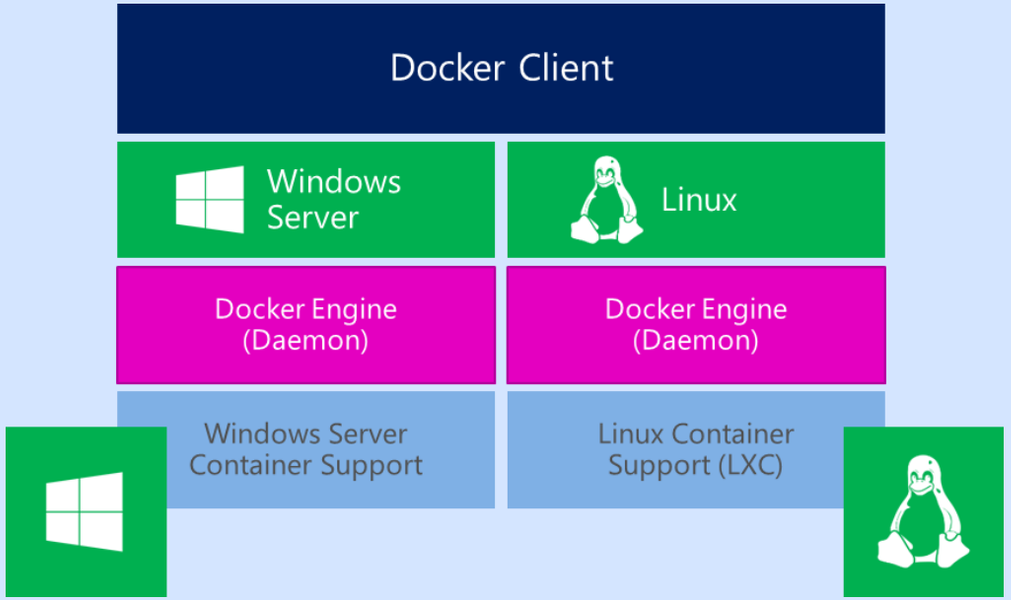
Containers are Managed with Docker

Docker available for Linux Containers since 2008

Open source engine that automates deployment of portable self-sufficient apps

Microsoft partnered with Docker to add support for Windows Containers in Windows Server 2016

Docker Experience maintained across both Linux and Windows platforms



Docker Overview

Docker is a Client-Server application used for managing Containers on a host

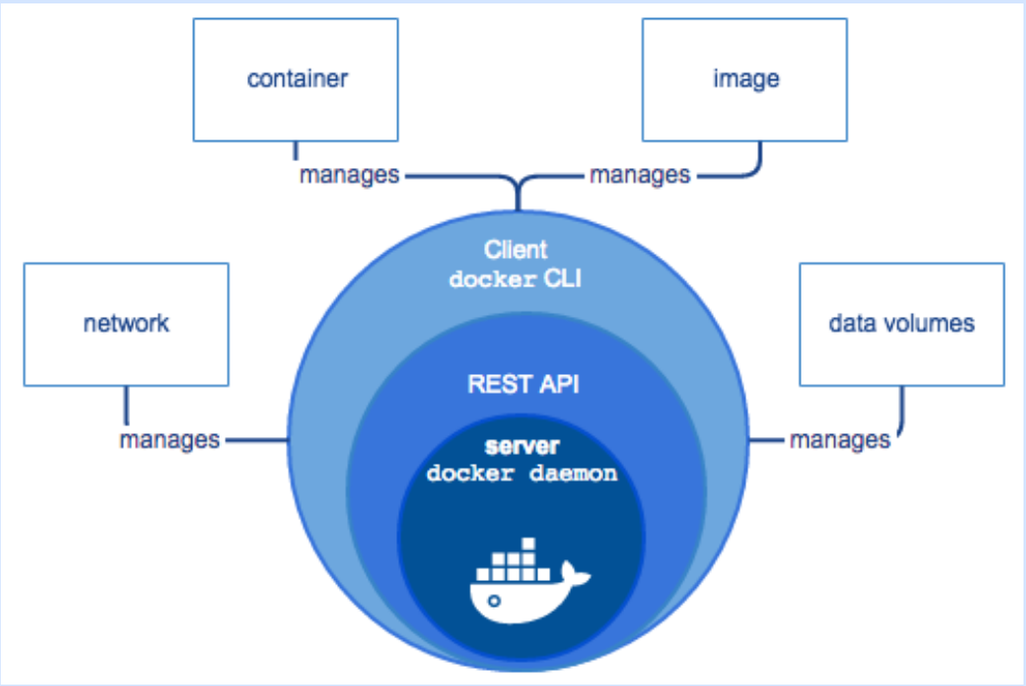
Docker is the vessel by which container images are packaged and delivered

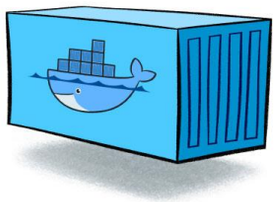
Docker Daemon: runs on a Container Host

- Builds, runs and distributes containers
- Client accesses via Sockets through REST API

Docker Client: interface to docker daemon

- Uses Command Line interface (i.e. docker.exe)
- Client can connect to a local or remote docker daemon





Docker Editions

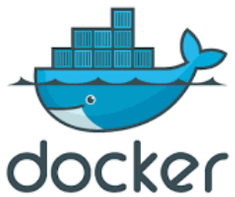
Docker Enterprise Edition (EE) for Windows Server

Three tiers (Basic, Standard, Advanced) **Click for Tier Detail**

- Basic available at no additional cost for Windows
- Technical support is aligned to the Microsoft support entitlement and provided by Microsoft

Docker Desktop (Windows)

- Previously named "Community Edition (CE) for Windows"
- Windows 10 Professional/Enterprise/Education 1607 or later
- Open Source / Free
- Must be installed with GUI
- Subset of the code delivered in EE
- Updates only available for most recent release



Docker Enterprise Edition for Windows Server

By Docker

Native Docker containers on Windows Server

Designed for enterprise development and IT teams who build, ship and run business critical applications in production. Required for orchestration.

Docker Desktop (Windows)

By Docker

The fastest and easiest way to get started with Docker on Windows

Ideal for individual developers and small teams developing containers on Windows 10. The containers are portable and can be moved to a Docker EE environment.

Docker EE Servicing

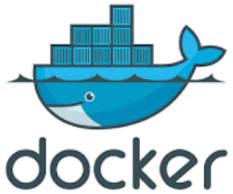
Docker Enterprise Edition (EE)

Updates available on Docker Hub via PowerShell

- DockerMsftProvider – Stable releases that have been validated by Microsoft
- DockerProvider – Stable and Edge releases

Support

- Stable releases supported for 24 months
- Hotfixes and patches for all supported releases



Administrator: Windows Powershell

```
PS C:\>Update-Module -Name DockerMsftProvider

PS C:\>
PS C:\>find-package -Provider DockerMsftProvider
-AllVersions

Name    Version          Source           Summary
----    -------          ------           -------
Docker 17.06.2ee-17    DockerDefault Contains..
Docker 18.03.1-ee-4    DockerDefault Contains..
Docker 18.09.0         DockerDefault Contains..

PS C:\>
PS C:\>install-package -Name docker
-ProviderName DockerMsftProvider
-RequiredVersion 18.09.0
-Update -Force

PS C:\>
PS C:\>Restart-service docker

PS C:\>
```

Docker Desktop Servicing

Docker Desktop (Windows)

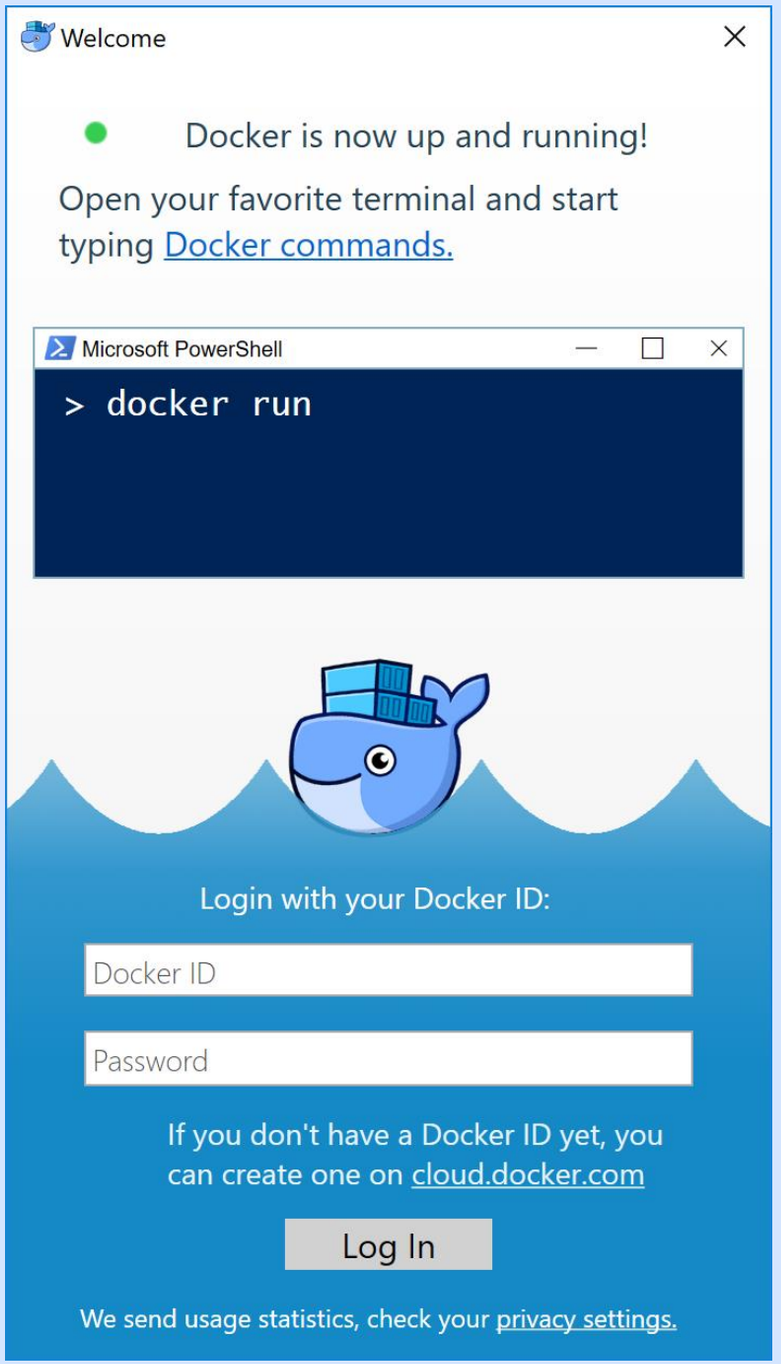
- Stable updates released every 6 months
- Edge updates released one or more a month

Update with the latest version by installing from

- Docker Store
- EXE on download.docker.com

Support

- Stable releases supported for 7 months
- Hotfixes and patches only for the most recent release



Microsoft

Getting Stated with Windows Containers

Overview of Container Hosts

Microsoft Services

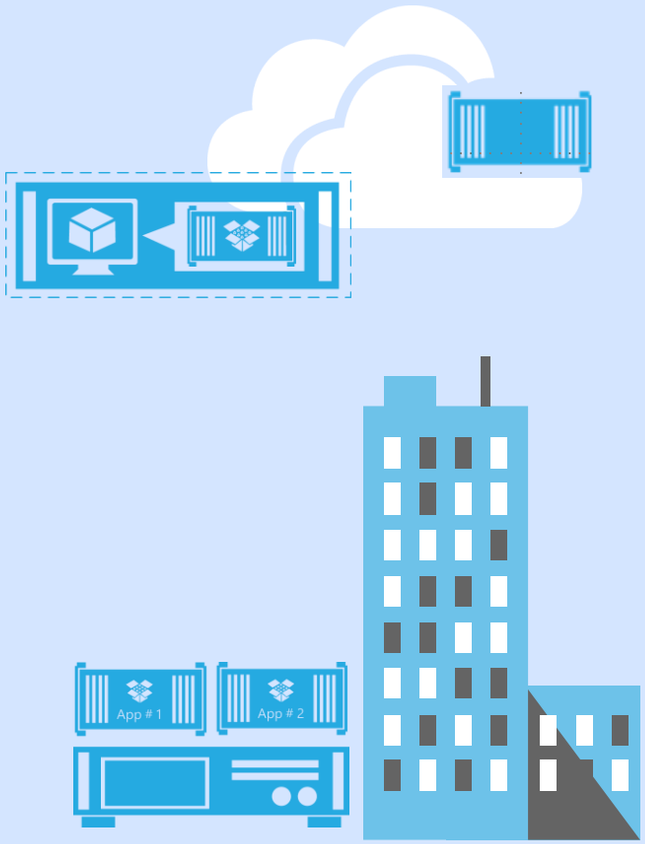
Container Host

Physical or Virtual system that hosts containers

A Windows-based Container Host is supported on

- Windows Server 2016 (or later)
- Windows 10 Professional or Enterprise, version 1607 (or later)
- The host can reside on-premises or in Azure

Note: Azure Container Instances are considered serverless, however the backend is Windows Server 2016



Container Host Licensing

Windows Server Standard Edition*

- Unlimited Windows Containers
- (2) Hyper-V Containers

Windows Server Datacenter Edition*

- Unlimited Windows Containers
- Unlimited Hyper-V Containers

Windows 10 Professional or Enterprise Edition

- (4) Hyper-V Containers

Feature	Datacenter Edition	Standard Edition
Core functionality of Windows Server	✓	✓
Operating System Environments (OSEs/Windows Servers containers with Hyper-V isolation)	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	✓	

Windows Server

Container Host Options in Azure

Azure Container Instances (ACI)

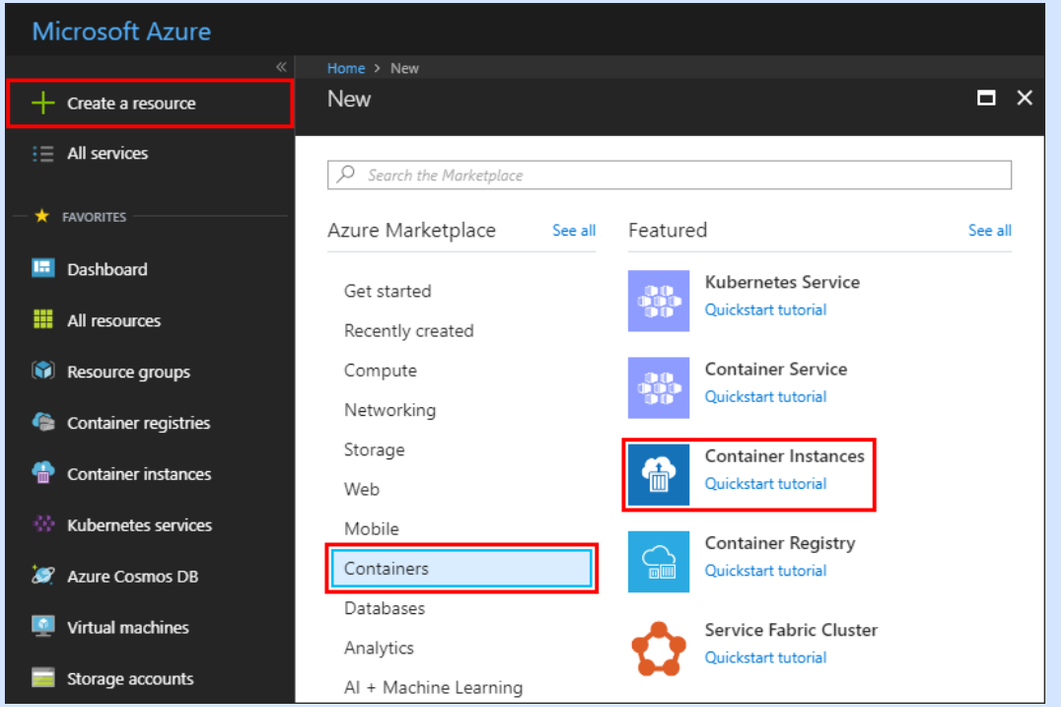
Serverless way to run both Windows and Linux containers in Azure

- On-demand computer service delivering rapid deployment of containers with no VM management and automatic elastic scale

Azure App Service

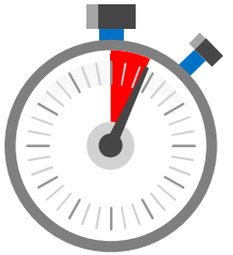
Support added for Windows Containers with **Web App for Containers** Public Preview

- Web App for Containers will deploy the containerized app with your preferred dependencies to production in seconds
- Just pull container images from Docker Hub or a private Azure Container Registry,
- The platform automatically takes care of OS patching, capacity provisioning, and load balancing



Azure: Azure Container Instances now available
Azure: Azure App Service Container Support

Container Host

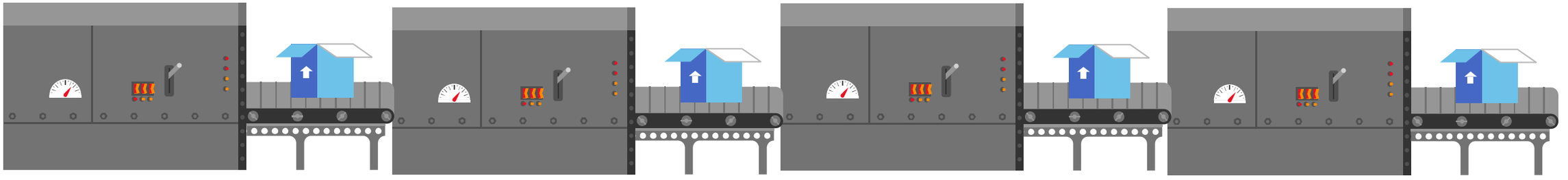


Anti-Virus Optimization

- Container Isolation filter (wcifs.sys) runs below the Anti-Virus filter range
- Many containers may depend on the same package layers
- This can result in AV scanning the same data in every container with potentially negative impact to Container Performance

Always Check with vendor to confirm if Anti-Virus Optimizations have been implemented in their product to support scanning Container Files

Optimizations included with Windows Defender by default



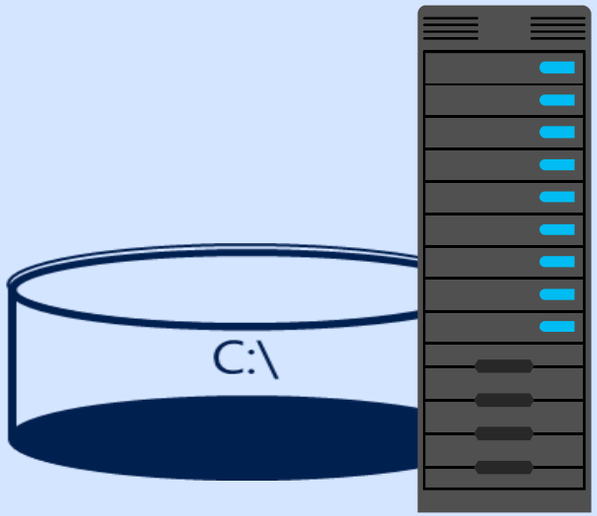
Container Host

Windows Container Requirements

Operating System must be installed on the C: drive of the **Container Host**

- Required to Support Isolated Layers

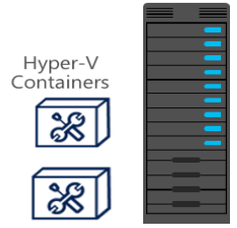
Note: not required for Hyper-V Containers



Windows Server

Container Host

Hyper-V Container Requirements



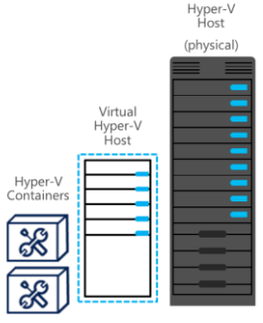
Windows Server

Physical Machine used for Container Host requires

- Enabling the Hyper-V Role on the Container Host

Virtual Machine used for Container Host requires

- Intel VT-x processor (Hyper-V Generation 2 VM)
- Virtualization enabled on VM (PowerShell cmdlet, disabled by default)
- Non-Dynamic (Static) Memory
- Minimum of 4gb Memory
- Minimum of 2 virtual processors
- MAC Address spoofing or NAT mode enabled



Windows Server

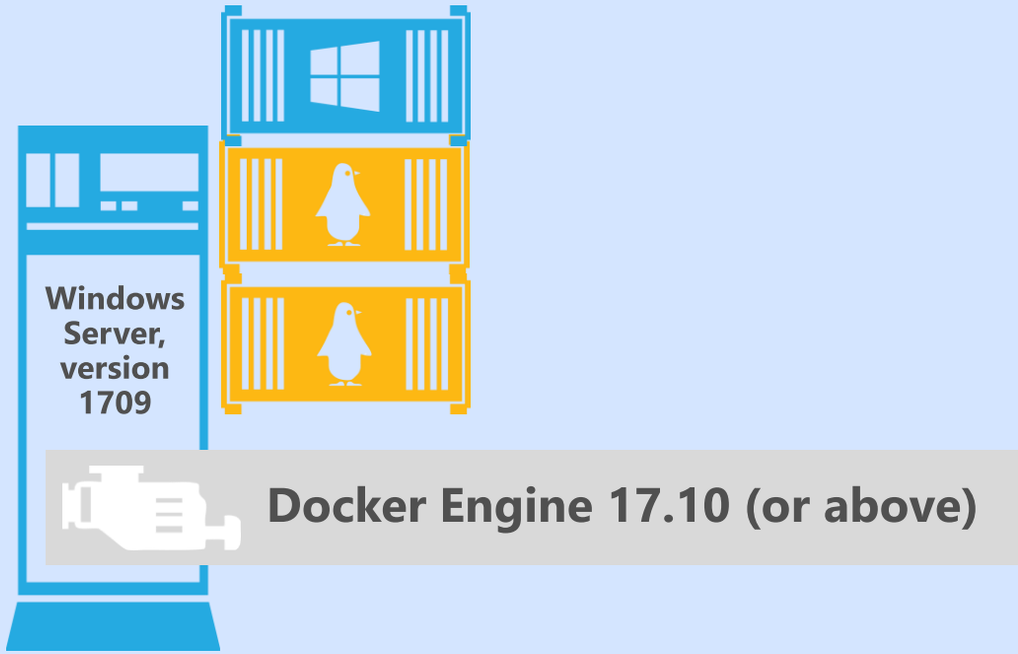
Click to view sample PowerShell Code
(Works in Slide Show View only)

Container Host

Linux Container Requirements

Support for Linux Containers is supported for Windows Server and Windows 10 version 1709 (or later)

- Hyper-V enabled (only supports Hyper-V Containers)
- Docker Engine version 17.10 (or above)
- Once installed set the following and restart docker
  - [Environment]::SetEnvironmentVariable("LCOW_SUPPORTED", "1", "Machine")
  - Restart-service docker



Windows Server, version 1709

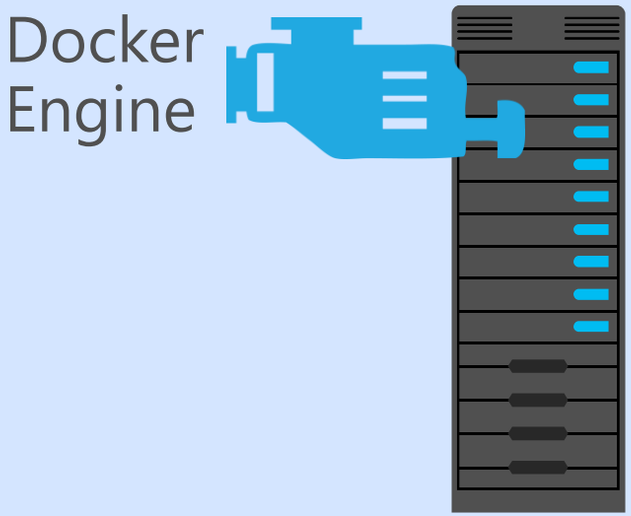
Container Host

Docker Engine Requirements

The Docker engine must be installed on the Container Host

- **Docker Enterprise Edition (EE)**: PowerShell cmdlets are used to download and install the docker package from the Powershell Gallery
- **Docker Desktop (Windows)**: Download and install with a GUI from Docker Store for download.docker.com on Windows 10

The Container Feature is automatically installed when the Docker engine is installed



Container Host

Procedure to install Docker EE with Internet Access

Install module for Provider from PowerShell Gallery

- DockerMsftProvider – for production builds
- DockerProvider – includes pre-release builds

Use **find-package** to list available versions

Install the **Docker** package

- Use -RequiredVersion to specify a version

Start the Docker service

Administrator: Windows Powershell

```
PS C:\>Install-Module -Name DockerMsftProvider
-Repository PSGallery -Force

PS C:\>
PS C:\>find-package -Provider DockerMsftProvider
-AllVersions

Name    Version          Source           Summary
----    -------          ------           -------
Docker 17.06.2-ee-17  DockerDefault Contains..
Docker 18.09.0        DockerDefault Contains..

PS C:\>
PS C:\>Install-Package -Name docker
-ProviderName DockerMsftProvider
-RequiredVersion 18.09.0

Are you sure you want to install software from Docker
Default?:
Y

PS C:\>
PS C:\>restart-service docker
```

Container Host

Procedure to install Docker EE without Internet access

Save **Docker** package using a system with Internet access

Extract the saved package from the zip file

Open a session on the Container Host without Internet Access and copy package to it

Enter the session, then register and start the docker service

Administrator: Windows Powershell

```
PS C:\>Save-Package -Provider DockerMsftProvider
-Name Docker -Path c:\packages
-RequiredVersion 18.09.0

Name   Version        Source         Summary
----   -------        ------         -------
Docker 18.09.0        DockerDefault Contains..

PS C:\>
PS C:\>expand-archive c:\packages\Docker-1-12-2-
cs.zip -DestinationPath c:\packages

PS C:\>
PS C:\>$session = New-PSSession <Container-Host>
PS C:\>copy-item -ToSession $session
-path "C:\packages\docker\*"
-Destination "C:\windows\system32" -Force

PS C:\>
PS C:\>Enter-PSSession $session
PS C:\>dockerd --register-service
PS C:\>Start-service docker
PS C:\>Exit-PSSession

PS C:\>
```

Container Host

Customization of Docker engine (optional)

Configuration options are available for configuring the Docker engine and are stored in

- C:\ProgramData\docker\config\daemon.json

For Example, change the default port

- Create a **daemon.json** file and configure Docker daemon to use port 2376 (uses 2375 by default)

```
{
"hosts": ["tcp://0.0.0.0:2376", "npipe://"]
}
```

- Open the firewall port
- Restart the docker service

Click for full list of configuration options
(Works in Slide Show View only)

Administrator: Windows Powershell

```
PS C:\>netsh advfirewall firewall add rule
name="Docker daemon “ dir=in action=allow
protocol=TCP localport=2376

PS C:\>
PS C:\>Restart-service docker

PS C:\>
```

Container Host

Configuration of Docker Service Proxy (optional)

To set proxy information for **docker search** and **docker pull**, create a Windows environment variable with the value of the proxy information

- **HTTP_PROXY** or
- **HTTPS_PROXY**

Set variable example →

Restart Docker service

Administrator: Windows Powershell

```
PS C:\>[Environment]::SetEnvironmentVariable
("HTTP_PROXY",“http://username:password@proxy:port/”,
[EnvironmentVariableTarget]::Machine)

PS C:\>
PS C:\>Restart-service docker

PS C:\>
```

Windows Server 2016 Editions

Feature	Datacenter	Standard
Core functionality of Windows Server	yes	yes
OSEs / Hyper-V containers	unlimited	2
Windows Server containers	unlimited	unlimited
Host Guardian Service	yes	yes
Nano Server installation option	yes	yes
Storage features including Storage Spaces Direct and Storage Replica	yes	no
Shielded Virtual Machines	yes	no
Software Defined Networking Infrastructure (Network Controller, Software Load Balancer, and Multi-tenant Gateway)	yes	no

Microsoft

Getting Stated with Windows Containers

Overview of a Container Image

Microsoft Services

Container Images

What is a Container Image?

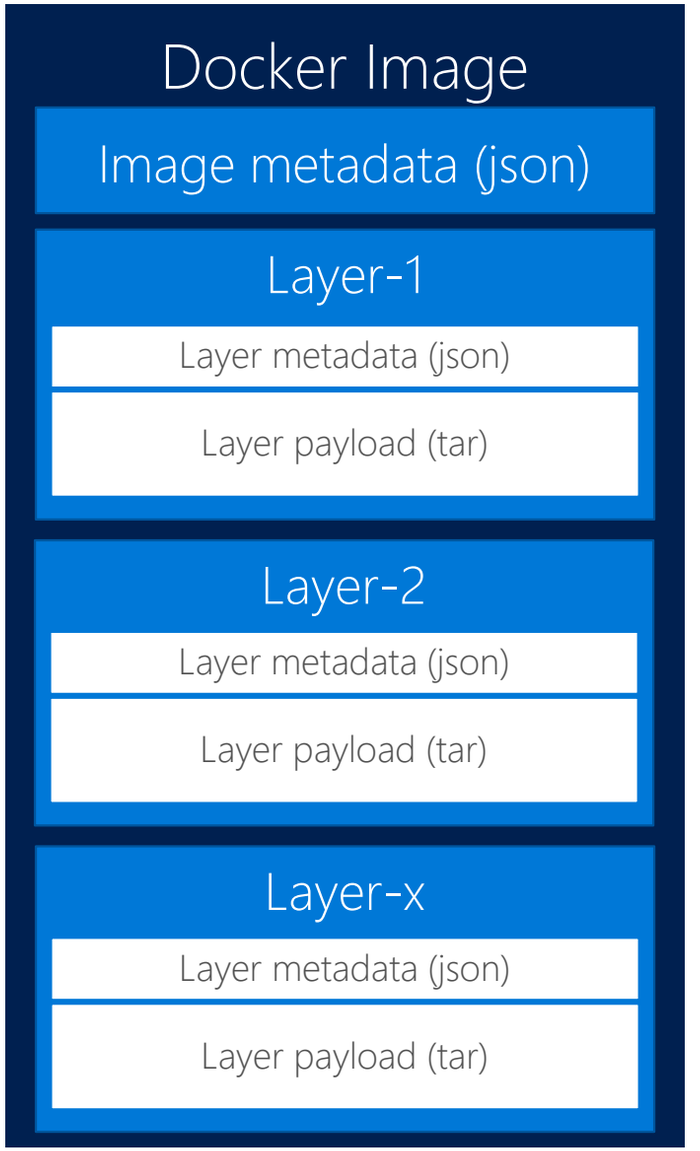
- Templates for containers
- Read-only
- File based

The Docker Implementation

- Metadata
- Layers

Creation

- Manually
- Automated with a Dockerfile
- Pulled from a registry



Container Images

What is a Container Image?

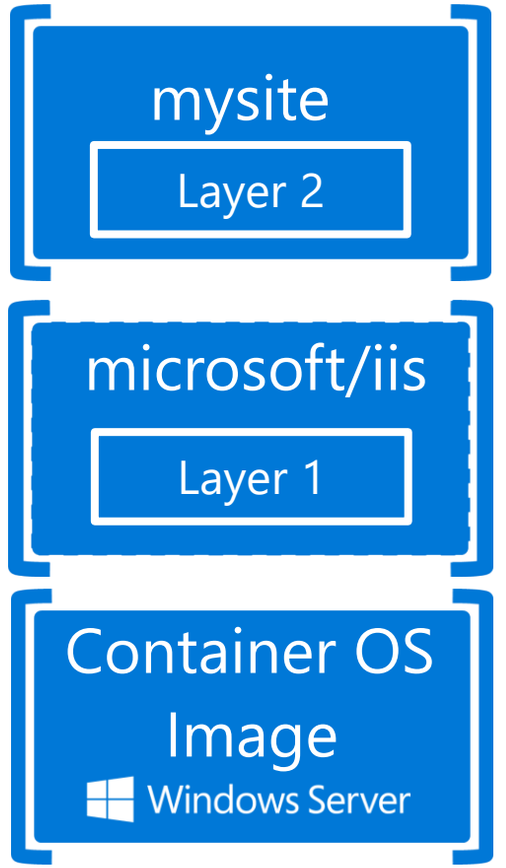
- Templates for containers
- Read-only
- File based

The Docker Implementation

- Metadata
- Layers

Creation

- Manually
- Automated with a Dockerfile
- Pulled from a registry



Container Layering

Container OS Image

- System Binaries
- C:\Windows

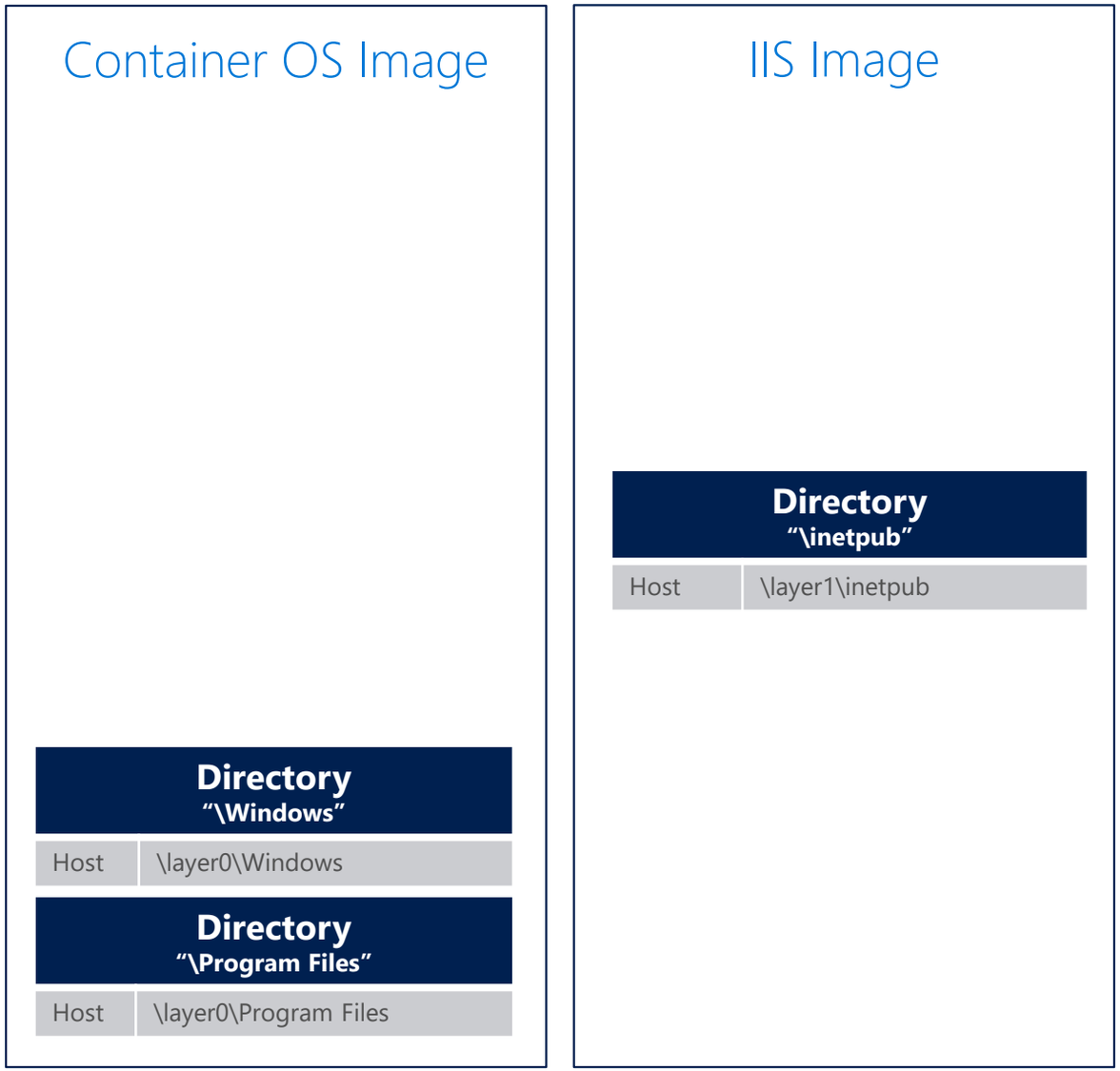
Choices

- Windows Server Core
- Nano Server

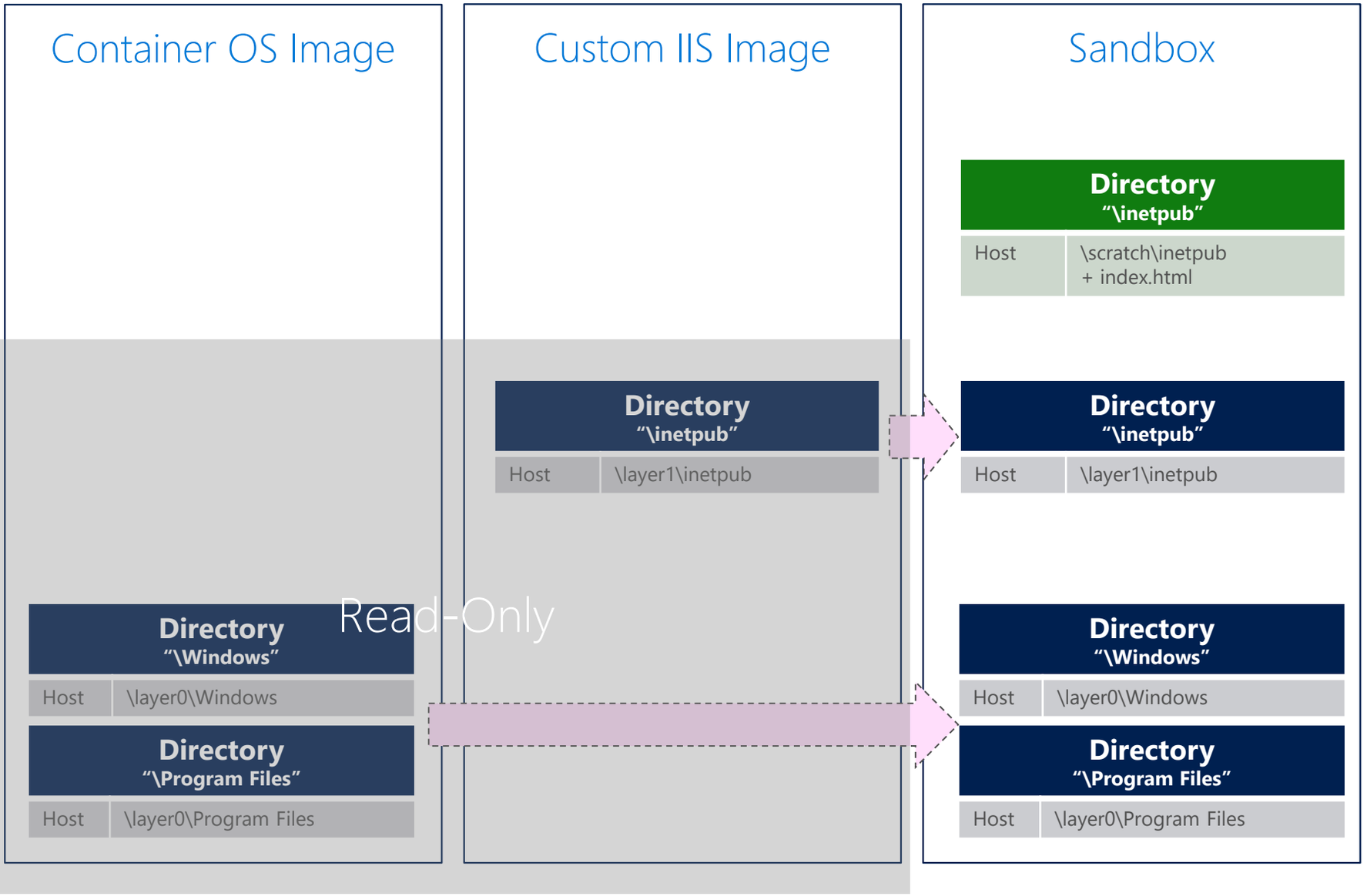
Availability

- Managed through ContainerImage provider
- Published by Microsoft

Container Layering



Running Container



Container Host Support

Windows Server 2016 Container Hosts supports

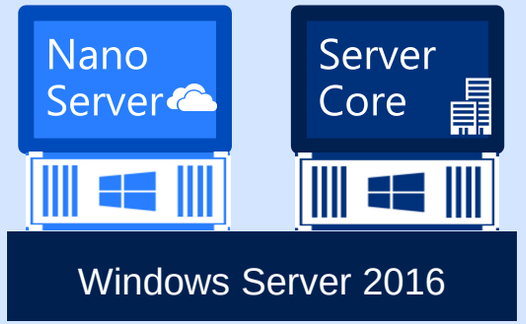
- Nano Server Based Container Images
- Server Core Based Container Images

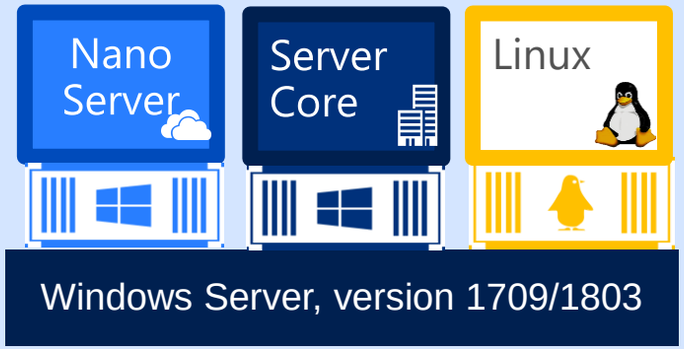
Windows Server, version 1709 and 1803 Container Hosts

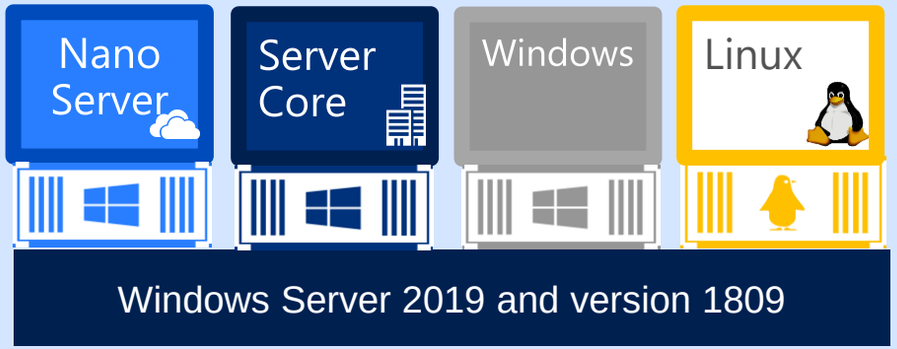
- Added support for Linux Based Container Images

Windows Server 2019 and Windows Server, version 1809 Container Hosts

- Added support for a new base image called “Windows” (described shortly)



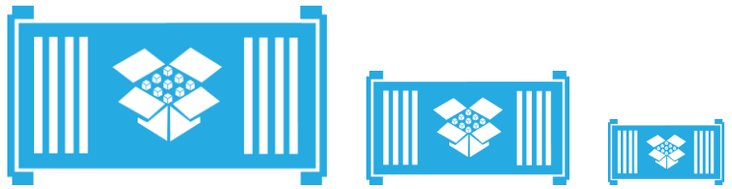




Server Core

Server Core Container Image

- Image size reduced by 50% in version 1709, another 30% in version 1803, and down to 1.5 GB in Windows Server 2019
- Unused optional features removed (can be added as needed) / List removed from version 1803 →
- Application compatibility improvements in each release
- Boot and run-time performance improved in each release



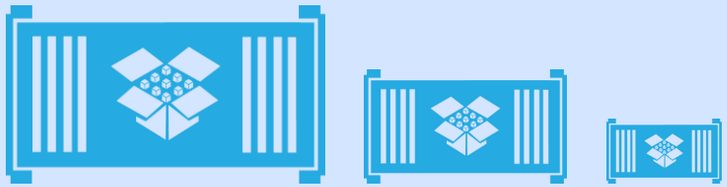
Windows Server, version 1709 and 1803

- AD Certificate Services
- AuthManager, Bitlocker
- BITS, CCFFilter, Containers
- CoreFileServer
- DataCenterBridging
- Dedup-Core
- DeviceHealthAttestationService
- DFSN and DFSR
- DirectoryServices
- DiskIo-QoS
- EnhancedStorage
- FailoverCluster
- File-Services
- FileServerVSSAgent
- FRS-Infrastructure , FSRM
- HardenedFabricEncryptionTask
- HostGuardian
- IdentityServer-SecurityTokenService
- IPAM, iSCSI, iSNS Service
- Licensing, LightweightServer
- Hyper-V, Group Policy
- Windows-FCI-Client-Package
- Windows-Subsystem-Linux
- MSRDC-Infrastructure
- MultipathIo , NetworkController
- NDES
- NetworkLoadBalancingFullServer
- NetworkVirtualization
- NFS Client and Server
- OnlineRevocationServices
- P2P-PnrpOnly, PeerDist
- Printing, QWAVE
- RasRoutingProtocols
- Remote-Desktop-Services
- RemoteAccess, ResumeKeyFilter
- RightsManagementServices-Role
- SBMgr-UI
- ServerCore-Drivers
- RSAT
- ServerMediaFoundation
- ServerMigration
- SessionDirectory
- SetupAndBootEventCollection
- ShieldedVMToolsAdminPack
- SMB1Protocol-Server
- SmbDirect
- SMBHashGeneration
- SmbWitness
- SNMP
- SoftwareLoadBalancer
- Storage-Replica-AdminPack
- Storage-Replica
- Tpm-PSH-Cmdlets
- UpdateServices-Database
- UpdateServices-Services
- UpdateServices-WidDatabase
- UpdateServices
- VmHostAgent
- VolumeActivation-Full-Role
- Web-Application-Proxy
- WebAccess
- WebEnrollmentServices
- Windows-Defender
- WindowsServerBackup
- WindowsStorageManagementService
- WINSRuntime
- WMISnmpProvider
- WorkFolders-Server
- WSS-Product-Package

Nano Server

Nano Server Container Image

- Image size reduced by over 80% in version 1709, and as of version 1803, image is below 100mb
- Optimized for .Net Core 2.0
- PowerShell Core, .NET Core and WMI no longer included by default in Container Image



Windows Server, version 1709 and 1803

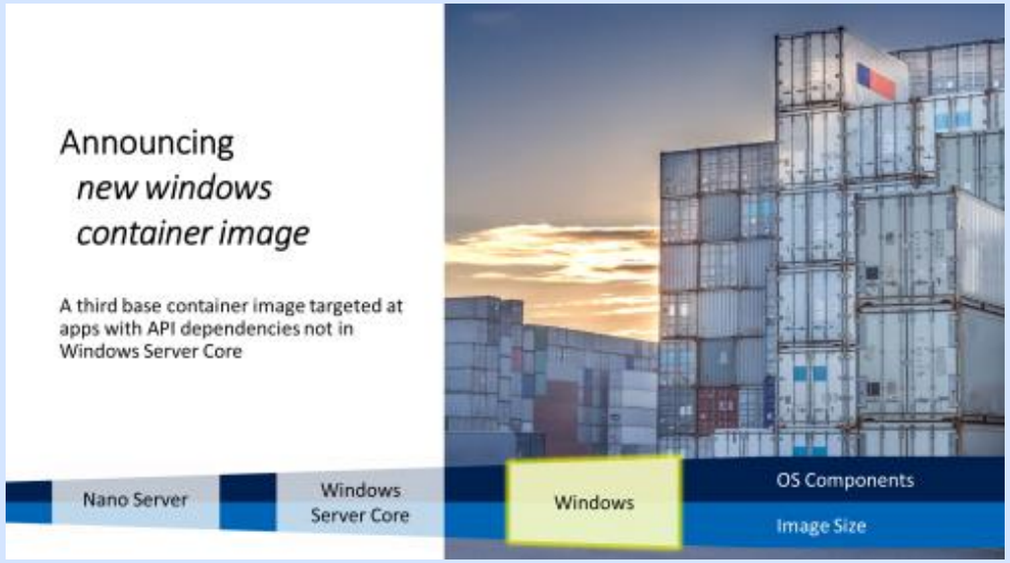
Windows

New! “Windows” Container Image

Based on customer feedback, Microsoft is introducing a third base image called “Windows” that supports applications with additional API dependencies.

Examples:

- Customers interested in moving legacy apps in containers, however some components were missing
- Customers interested in leveraging containers to run automated User Interface tests and needed graphic capabilities like DirectX



Windows Server 2019

MS Docs: aka.ms/WindowsContainer

Windows Base OS Image Summary

Nano Server	Windows Server Core	Windows
Born in the cloud applications	App Compatibility	Automation Workloads
.NET core Support	Full .NET framework support	Carries most Windows OS components
94 MB Image Size	1.4 GB Image Size	3.5 GB Image Size

Container Image Tags

Understanding Tags

Windows Server 2016

- Latest Server Core version: ltsc2016
- Latest Nano Server version: sac2016
- Build specific: 10.0.14393.<*revision*>

Windows Server, version 1709, 1803 or 1809

- Latest version: 1709, 1803, 1809
- Build specific: 1709_KB<*number*>, 1803_KB<*number*>, 1809_KB<*number*>

Windows Server 2019

- Latest Server Core version: ltsc2019
- Latest Nano Server or Windows version: sac2019
- Build specific: 1809_KB<*number*> or 10.0.17763.<*revision*>

***1/26/2019 (underscores changed to hyphens for consistency)*

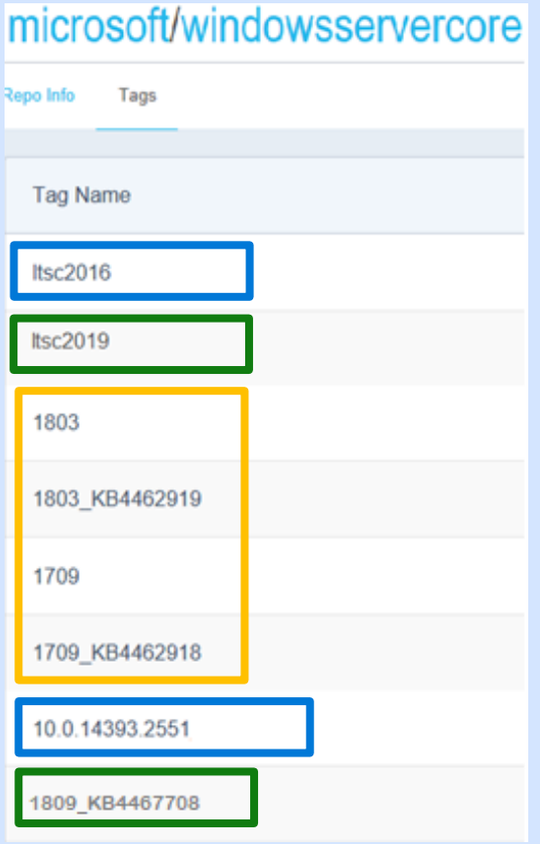
Understanding OS Version Format

<Major>.<Minor>.<Build>.<Revision>

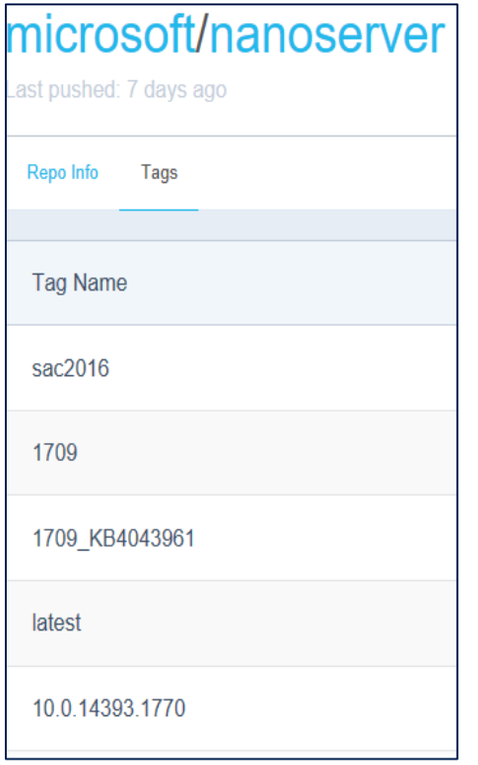
- *Build is the OS version (i.e. Server 2016)*
- *Revision is the monthly Patch version (i.e. KB)*

Examples with April 2018 update

- Windows Server 2016: 10.0.14393.2214
- Windows Server, version 1709: 10.0.16299.371



Find Available Container Images



Use Docker Hub to find Tags for required Build and Revision

Use **WINVER** (GUI) or **VER** (cmd prompt only) or a **Registry query** (example below) to obtain Build and Revision of Container Host

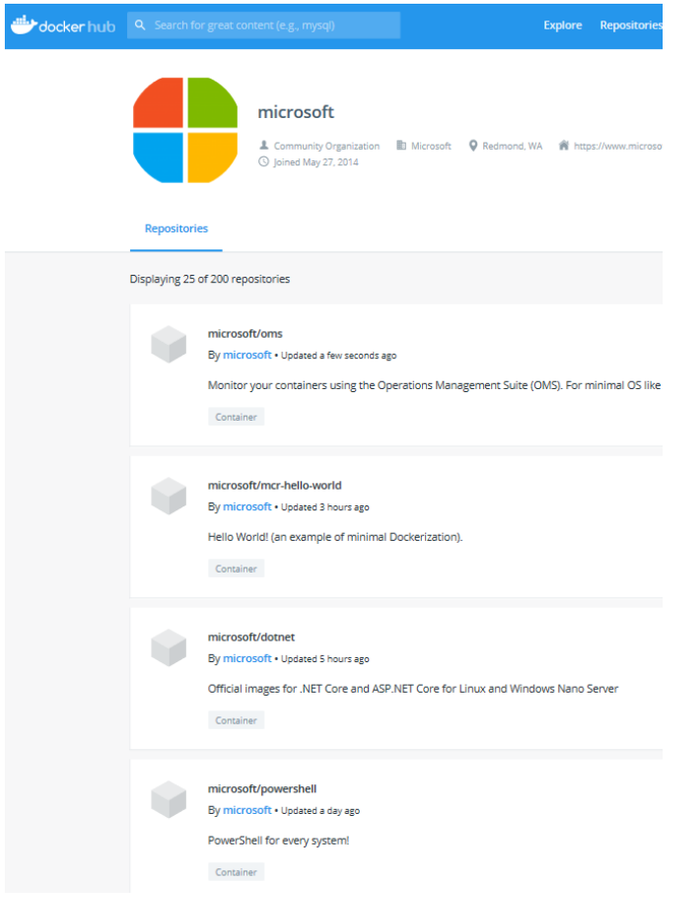
```
Get-ItemPropertyValue -Path "HKLM:\Software\Microsoft\Windows NT\CurrentVersion\" -Name "UBR“
```

Reminder: When using Windows Containers on a Windows Server 2016 or Windows 10, version 1607 Container Host the OS Build and Revision of the Container must match.

Leverage Docker hub to find tags

- New: https://hub.docker.com/search?q=microsoft&type=image&image_filter=store
- Original: https://hub.docker.com/r/microsoft

Find Available Container Images



Use Docker Hub to find Container Images (original hub interface)

https://hub.docker.com/r/microsoft

- Includes syntax for docker pull command
- Includes tag information (which we will go over in detail later)

Demo: Navigate to some of these web pages and review the Overview information

Note **Important Update** listed at top of each page regarding new publisher page (will discuss on next slide).

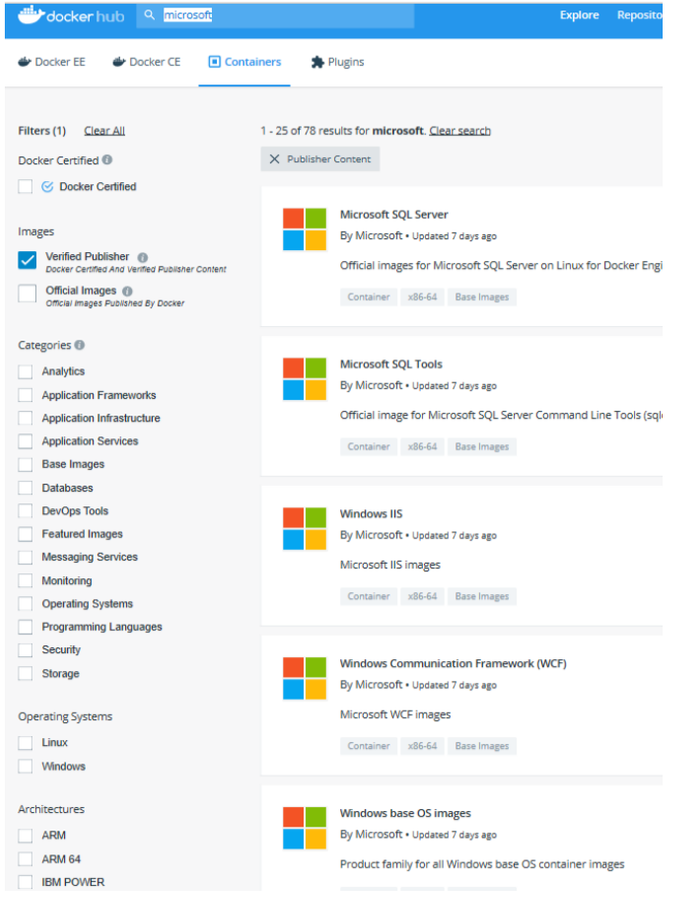
Container Image Examples	Link to Tag List
Windows Server Core	https://hub.docker.com/r/microsoft/windowsservercore/
Nano Server	https://hub.docker.com/r/microsoft/nanoserver/
Windows	https://hub.docker.com/r/microsoft/windows
IIS	https://hub.docker.com/r/microsoft/iis/
Windows Server Core Insiders	https://hub.docker.com/r/microsoft/windowsservercore-insider/
Microsoft SQL Server	https://hub.docker.com/r/microsoft/mssql-server-windows/

Find Available Container Images

Use Docker Hub to find Container Images (new publisher page)

https://hub.docker.com/search?q=microsoft&type=image&image_filter=store

Demo: Navigate to web pages and review the docker pull details



Container Image Examples	Link to Tag List
Windows base OS Images	https://hub.docker.com/_/microsoft-windows-base-os-images
Windows Server Core	https://hub.docker.com/_/microsoft-windows-servercore
Nano Server	https://hub.docker.com/_/microsoft-windows-nanoserver
Windows	https://hub.docker.com/_/microsoft-windowsfamily-windows
Windows IIS	https://hub.docker.com/_/microsoft-windows-servercore-iis
Windows Server Core Insider	https://hub.docker.com/_/microsoft-windows-servercore-insider
Microsoft SQL Server	https://hub.docker.com/_/microsoft-mssql-server

Cloud Blogs: Improved Discovery Experience

Container Servicing

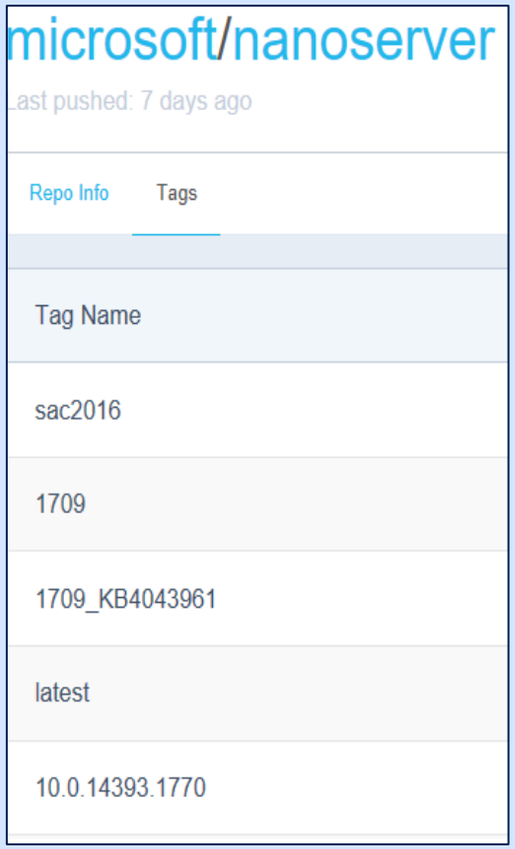
Container Images Updated Monthly

Microsoft releases Container Images monthly which are viewable on docker hub

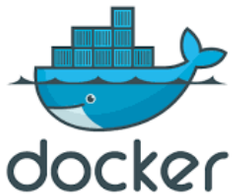
Images correspond with monthly security and non-security update releases

Container Images use **Tags** to indicate version

- On Server Core and Nano Server based images the tag indicates build version and patch level



http://hub.docker.com/u/microsoft

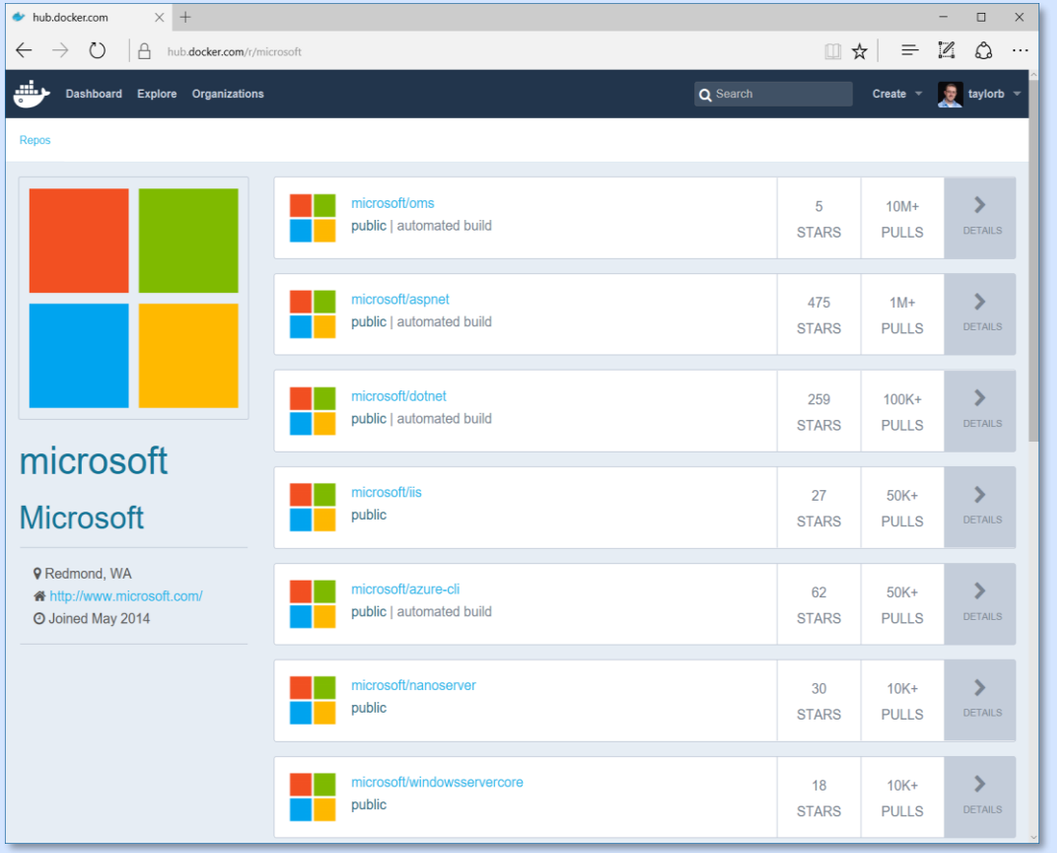


New! Microsoft Container Registry

New! Microsoft Container Registry (MCR)

Microsoft plans to migrate all microsoft images from microsoft/* to mcr.microsoft.com/*

- Microsoft is working with Docker to provide a consistent experience with Docker Hub
- Microsoft Images from both registries will be viewable from **Docker Hub**
- The Docker Hub landing pages have been updated to reflect the new pull location



Current Status for Microsoft Images (Nov 2018)

Current Status for Microsoft Images (Nov 2018)

Hosted as follows

- Windows Server, 1803 (and earlier) Container Images are hosted on the Docker Hub

  docker pull microsoft/<*image*>:<*tag*>

- Windows Server 2019 and Windows Server, version 1809, Container Images are hosted in MCR

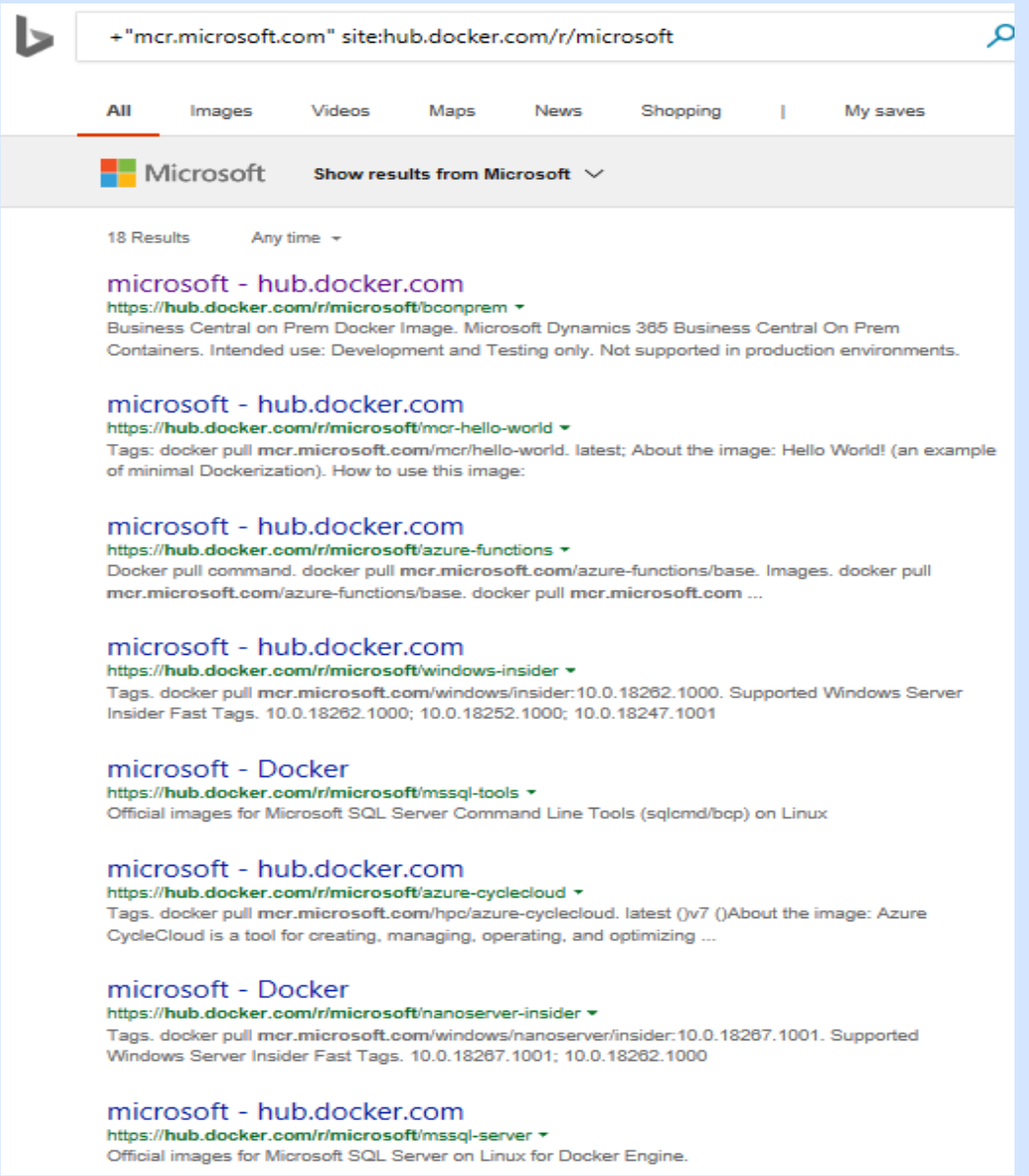
  docker pull mcr.microsoft.com/<*image*>:<*tag*>

Documentation for Container images from both registries can be viewed a

https://hub.docker.com/r/microsoft

Search for images stored in MCR from bing.com

- +"mcr.microsoft.com" site:hub.docker.com/r/microsoft



Overview of Container Host Support of Container Images by OS

Container Host OS**	Container Image OS				
	Windows Container - By Tag / OS Version				Linux Container
	sac2016, ltsc2016 10.0.14393	1709 10.0.16299	1803 10.0.17134	sac2019,ltsc2019,1809 10.0.17763	
Windows 10 Ent/Pro 1609, 1703	Hyper-V	No	No	No	No
Windows 10 Ent/Pro 1709	Hyper-V	Hyper-V	No	No	Hyper-V
Windows 10 Ent/Pro 1803	Hyper-V	Hyper-V	Hyper-V	No	Hyper-V
Windows 10 Ent/Pro 1809	Hyper-V	Hyper-V	Hyper-V	Both (next slide)	Hyper-V
Windows Server 2016	**Both**	No	No	No	No
Windows Server, version 1709	Hyper-V	**Both**	No	No	Hyper-V
Windows Server, version 1803	Hyper-V	Hyper-V	**Both**	No	Hyper-V
Windows Server 2019 / version 1809	Hyper-V	Hyper-V	Hyper-V	**Both**	Hyper-V
Azure Container Instance***	Windows	No	No	Windows	Yes

LTSC

LTSC

- Container Images are <u>not</u> backwards compatible with Container Hosts running an earlier OS
- **Azure Virtual Machine - Hyper-V Containers are only supported on VM sizes that include support for nested virtualization (Dv3 or Ev3 series available as of July 2017)
- ***Azure Container Instances leverage Windows Server 2016 on the backend

MS Docs: Version Compatibility

Windows Container Support on Windows 10

Starting in Windows 10, version 1809 with the October 2018 cumulative update support was added to allow Process isolation

- Previously, Windows Containers (which use process isolation) were not supported on Windows 10
- Only Hyper-V Containers were supported using the parameter "--isolation=hyperv"
- Now Windows Containers are supported for test and development only!
- Requires
  - "--isolation=process" parameter
  - Windows 10 build 17763 or later
  - Docker Engine 18.09 or later

Windows 10, version 1809 with Oct 2018 Update

MS Docs: Windows 10 and Windows Containers

How to Download a Container Image to a Container Host with Internet Access

Use **docker pull** to download images from the Microsoft Repository on Docker Hub

Specifying a Tag is helpful for keeping track of image build versions in the local repository

Tag Examples:

- Tag 10.0.14393.2551 is Oct 2018 Monthly Cumulative Update for Windows Server 2016
- Tag 1803_KB4462919 is Oct 2018 Monthly cumulative update for Windows Server, version 1803

**Tag changes to support new Microsoft Container Registry for Windows Server 2019 and version 1809

- docker pull mcr.microsoft.com/windows/nanoserver:1809_KB4464330

Administrator: Windows Powershell

```
PS C:\>docker pull microsoft/nanoserver:10.0.14393.2551

10.0.14393.2551: Pulling from microsoft/nanoserver
bce2fbc256ea: Downloading [==>] ..
6f2071dcd729: Pull complete
Status: Downloaded microsoft/nanoserver:10.0.14393.2551

PS C:\>
PS C:\>docker pull microsoft/nanoserver:1803_KB4462919

1803_KB446919: Pulling from microsoft/nanoserver
bce2fbc256ea: Downloading [==>] ..
6f2071dcd729: Pull complete
Status: Downloaded microsoft/nanoserver:1803_KB446919


PS C:\>
PS C:\>docker pull
mcr.microsoft.com/windows/nanoserver:1809_KB4464330

10.0.17763.55: Pulling from microsoft/nanoserver
bce2fbc256ea: Downloading [==>] ..
6f2071dcd729: Pull complete
Status: Downloaded microsoft/nanoserver:1809_KB4464330

PS C:\>
```

How to Download a Container Image to a Container Host without Internet Access

Once a container image is downloaded to a system with Internet Access (previous slide), then

- Save the image using docker save
- Copy to systems without Internet Access
- Load the image using docker load

Recommendation: Develop a naming scheme for .docker files, so that they are easy to identify

Note: .docker files can be stored on a File Share

Administrator: Windows Powershell

```
PS C:\>docker save
-o c:\images\nanoserver_14393_2551.docker
microsoft/nanoserver:10.0.14393.2551

PS C:\>
PS C:\>$session = New-PSSession <Container-Host>
PS C:\>copy-item -ToSession $session
-path "C:\images"
-Destination "C:\" -Recurse -Force

PS C:\>
PS C:\>Enter-PSSession $session
PS C:\>docker load
-I c:\images\nanoserver_14393_2551.docker

PS C:\>
```

How to List Container Images in the Local Repository of the Container Host

Run **docker images** (example below)

- Remember that generic tags do not specify build revision

  latest, ltsc2016, sac2016, 1709, 1803, 1809, ltsc2019, sac2019

- To determine Build Revision with a generic tag, open an interactive session with the container and obtain from the registry

Click for demo of interactive session to obtain Build Revision of container with generic tag
(Works in Slide Show View only)

Administrator: Windows Powershell

```
PS C:\>docker images

REPOSITORY              TAG                                 IMAGE ID            CREATED             SIZE
microsoft/nanoserver    10.0.14393.2551                     4c872414bf9d        4 weeks ago         1.17GB
microsoft/nanoserver    1709_KB4462918                      e2bf669ca762        4 weeks ago         336MB
microsoft/nanoserver    1803_KB4462919                      f3a745168634        4 weeks ago         358MB
microsoft/nanoserver    10.0.17763.55                       a79befa5dcde        4 months ago        232MB
microsoft/nanoserver    sac2016                             5c872414cf9d        4 weeks ago         1.17GB
microsoft/nanoserver    1803                                e3a746168634        4 weeks ago         358MB
Microsoft/iis           20181009-nanoserver-sac2016         947ef07fa481        4 weeks ago         1.29GB
Microsoft/iis           20181009-windowsservercore-ltsc2016

PS C:\>
```

Demonstration: *Building and Running IIS Server Container*

Build IIS Container Image using Dockerfile

Run IIS Container

Microsoft

Getting Stated with Windows Containers

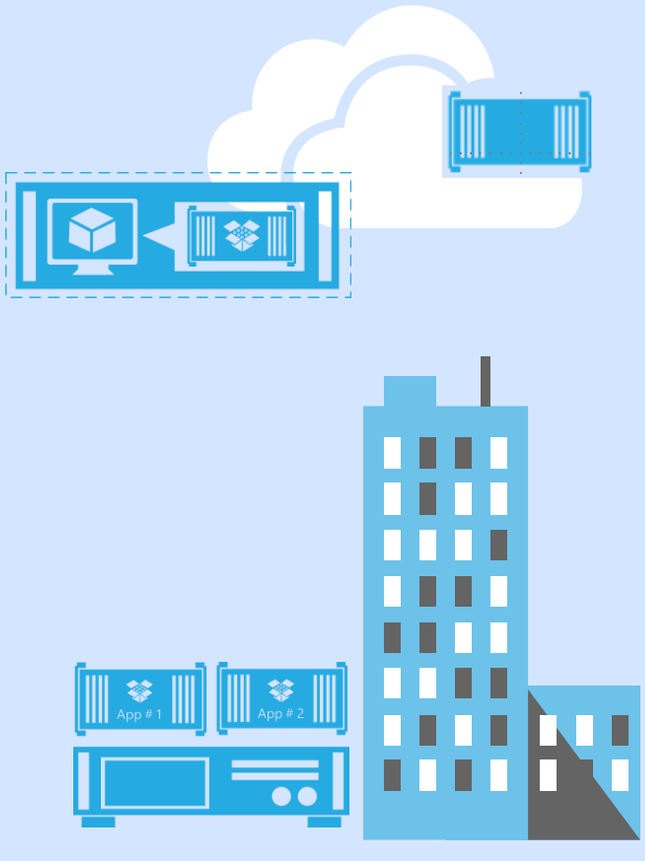
Create and Run Simple Container

Microsoft Services

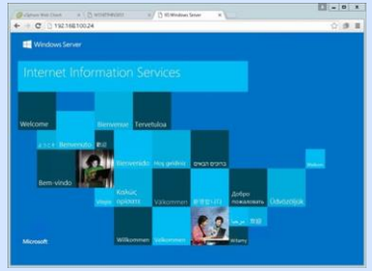
Creating Containers

Overview

- Create and run a simple Windows Container with the IIS Role
- Create and run a simple Hyper-V Container running Ping command
- Example of simple commands to manage Containers



Create and Run



Create a Simple Windows Container with the IIS Role

docker run [OPTIONS] IMAGE [COMMAND] [ARG...]

IMAGE create a writeable container layer over

microsoft/iis:20181009-windowsservercore-ltsc2016

[OPTIONS]

(-d) Run container as a background service

(-p 8080) map port 80 of the container host to port 80 of the container

(-name DemoFirst) this is case sensitive

Summary:

- The Container Host does not have the IIS Role installed
- All traffic to port 80 of the Container Host is redirected to the IIS server within the DemoFirst Container
- The container can be customized with a web page, stored in a registry and used by multiple container hosts

Administrator: Windows Powershell

```
PS C:\>docker run -name DemoFirst -d -p 8080
microsoft/iis:20181009-windowsservercore-ltsc2016
bB5df52e5b4fc422931d067d9c5f44ffdd545c3192c817626de17…

PS C:\>
PS C:\>docker ps
CONTAINER ID  IMAGE           COMMAND CREATED    Name
b5df52e5b4fc  microsoft/iis.. C:\\w.. 15 secs.. DemoFirst

PS C:\>
```

Create and Run

Create a Simple Hyper-V Container running Ping

docker run [OPTIONS] IMAGE [COMMAND] [ARG...]

IMAGE Create a writeable container layer over
microsoft/nanoserver:10.0.14393.2551

[OPTIONS]

(-d) run container as a background service

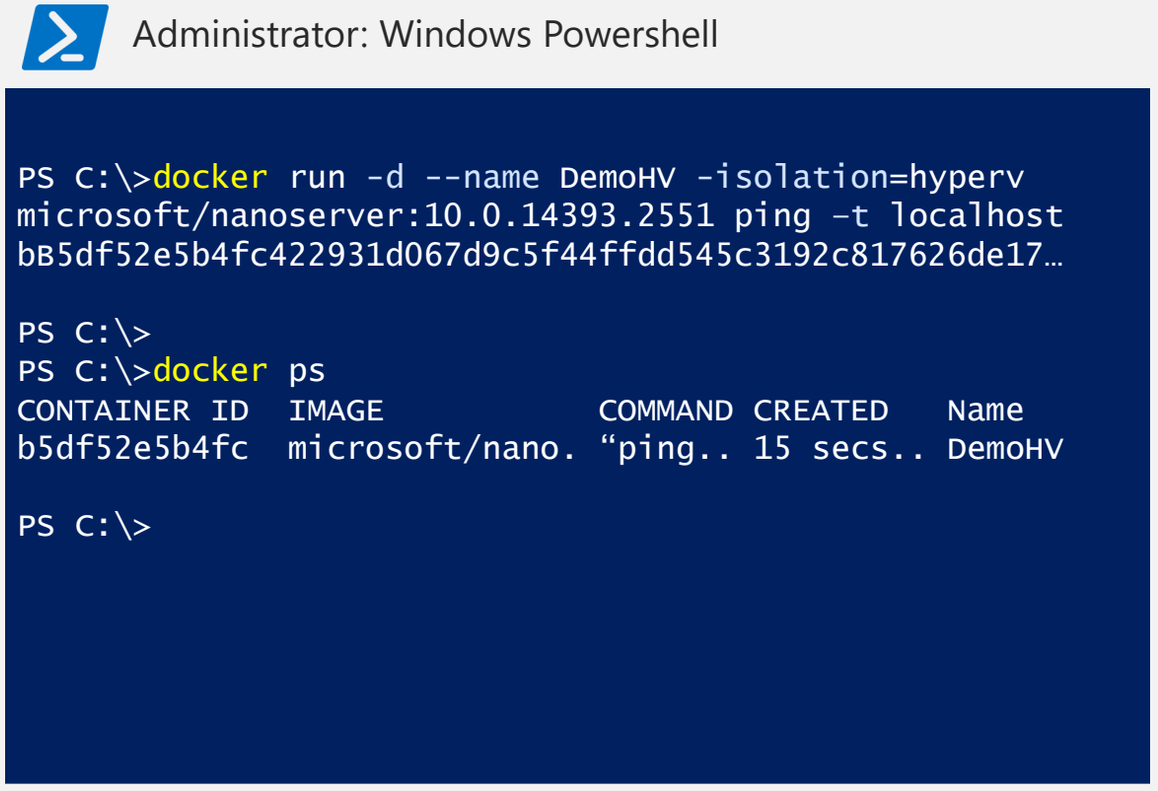
(--isolation=hyperv) Hyper-V Container

[COMMAND] Program to start in Container

(ping –t localhost) run ping continuously in Container

Summary:

- Ping.exe will run continuously within the Container
- In a real world scenario, this would be an application that was added to the container during customization



Manage

Examples of Simple Commands to manage containers

Note: docker is case-sensitive

- List Running Containers
- Stop a running container
- List stopped and running containers
- Delete a container

```
#List running Containers
Docker ps

#Stop a running Container
Docker stop DemoFirst

#List stopped and running containers
Docker ps -a

#Delete a Container
Docker rm DemoFirst
```

Microsoft

Getting Stated with Windows Containers

Customize Container Images

Microsoft Services

Container Customization

Deployment Options

Interactive

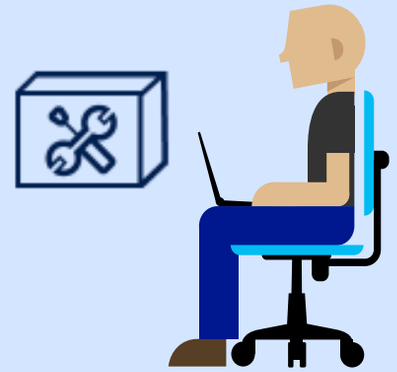
- Open a Container in an Interactive Session, manually add files, applications, scripts, registry entries, etc. and then commit changes

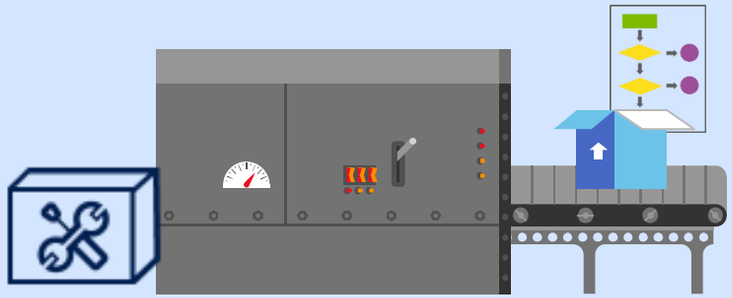
Automated

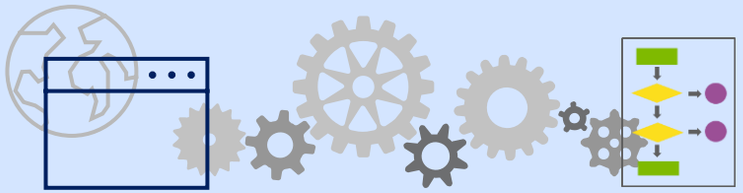
- Create a DockerFile that includes instructions to automatically customize a Container and commit changes

Conversion

- Use Image2Docker to convert an existing application to a DockerFile







Interactive Session

Open a DOS command prompt and open a container in an interactive session

docker run --name <*name*> -it <*image*> powershell.exe

Customize container by adding files, editing the registry, etc.. And Exit the session

Commit the changes in Layer 3 (i.e. Sandbox) to Layer 2

docker commit <*Container ID*><*new name*>

Run the new container and then remove it on exit

docker run –rm <*new name*>

Click for Interactive Session Demo
(Works in Slide Show View only)

Dockerfile – Build ASP.NET 4.5 Container Image

- Leverage IIS Container Image

  FROM microsoft/iis:latest

- Install .NET and ASP.NET 4.5

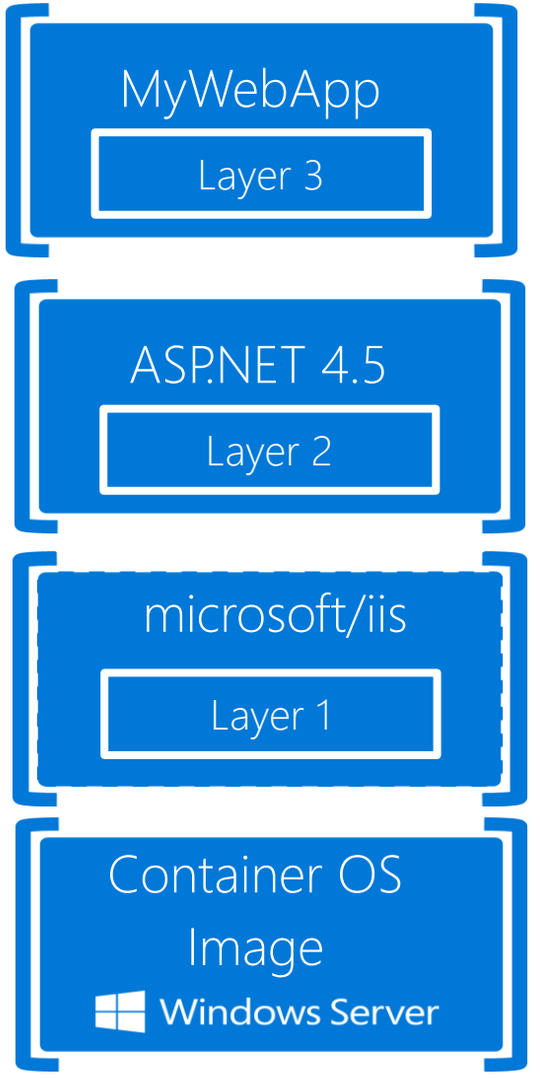
  RUN Install-WindowsFeature Web-Asp-Net45

- Copy MyWebApp to Container

  COPY MyWebApp MyWebApp

- MyWebApp IIS WebApplication

  RUN New-Website -Name 'guidgenerator' -Port 80 \
  -PhysicalPath 'c:\WebAppLegacy' -ApplicationPool '.NET v4.5'



Demonstration: *Package ASP.NET 4.5 Web Application as Container*

Containerized ASP.NET 4.5 Web Application

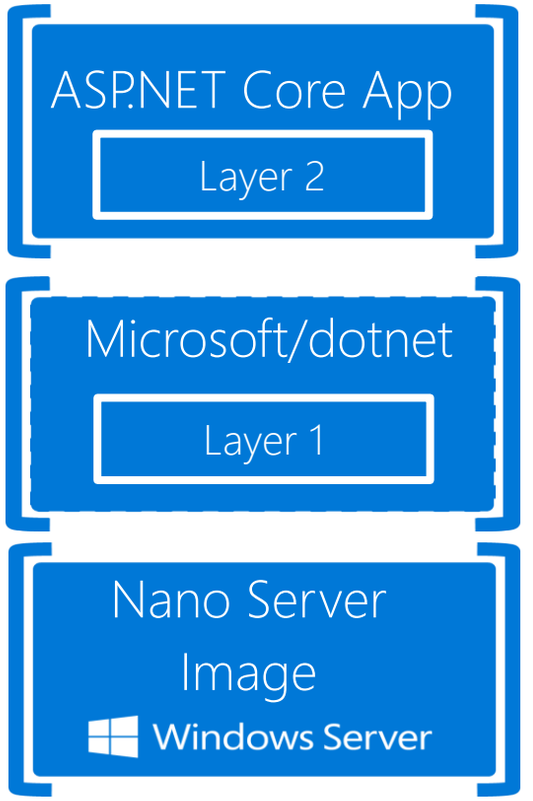
Run ASP.NET 4.5 Web Application as Container

Dockerfile – Build ASP.NET Core Container Image

- Leverage IIS Container Image
  FROM microsoft/dotnet:nanoserver

- Copy ASP.NET Core App to Container
  COPY published ./

- Entrypoint set to Application
  ENTRYPOINT ["dotnet", "mywebapp.dll"]



Demonstration: *Package ASP.NET Core Web Application as Container*

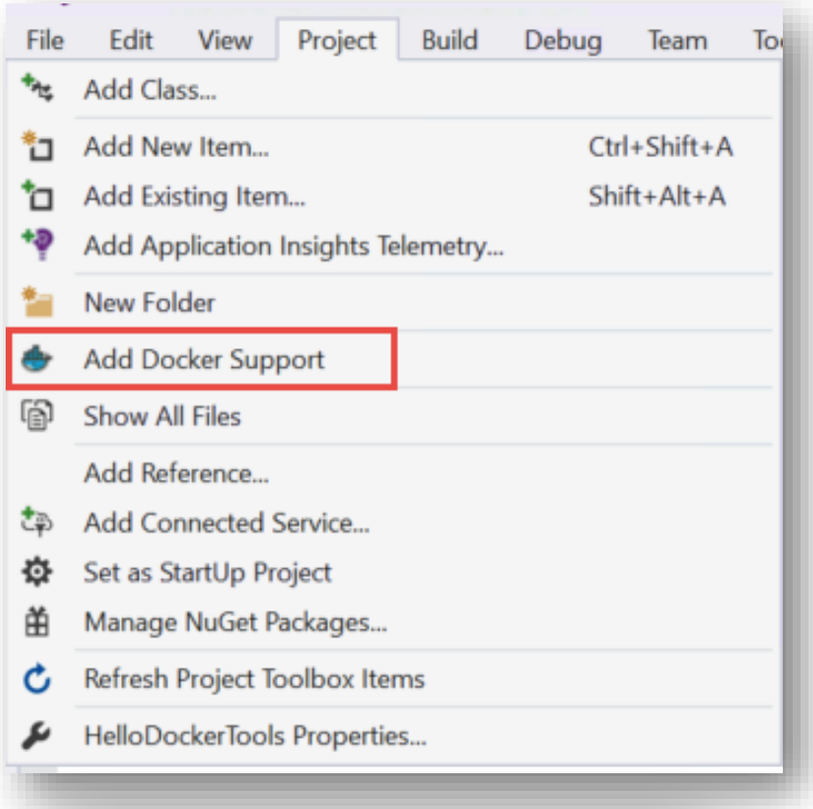
Containerized ASP.NET Core Web Application

Run ASP.NET 4.5 Core Application as Container

Visual Studio Tools for Docker

Microsoft Visual Studio 2017 provides integrated developer experience with Docker.

Building, *Debugging*, and *Running* .NET Framework and .NET Core web and console applications using Windows and Linux containers.



Demonstration: *Visual Studio 2017 and Docker*

Building ASP.NET Core Application using Visual Studio

Debugging ASP.NET Core Application using Visual Studio

Convert Application to Dockerfile

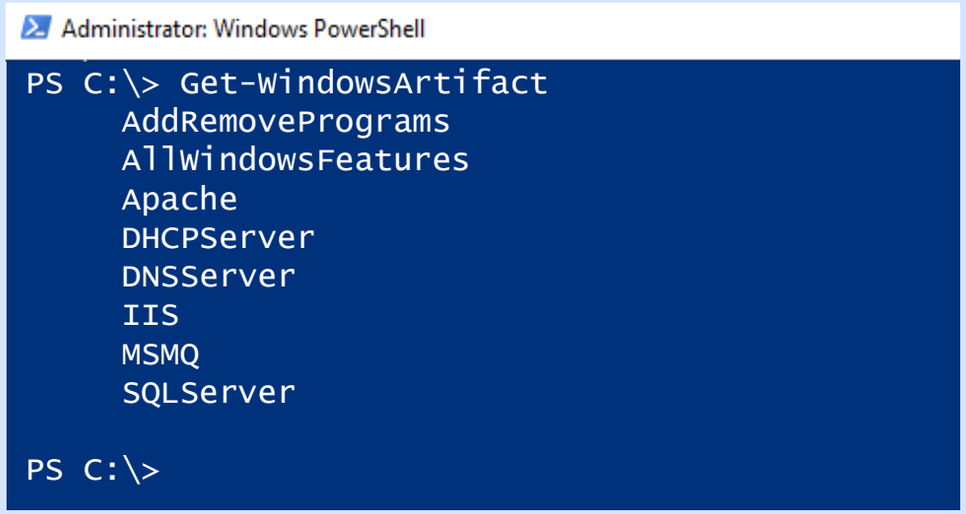
ConvertTo-Dockerfile: Open Source PowerShell Script

Targets WIM, VHD, VHDX and Live Servers

Supports conversion of the following Artifacts

- Roles and Features
- Add/Remove Programs
- Internet Information Services (IIS)
  - HTTP Handlers in IIS Configuration
  - IIS Websites and filesystem paths
  - ASP.Net web application
- Microsoft SQL Server Instances
- Apache Web Server

```
#Install Image2Docker from PowerShell Gallery
install-module -name Image2Docker
import-module -name Image2Docker

#List supported Artifacts
Get-WindowsArtifact
```

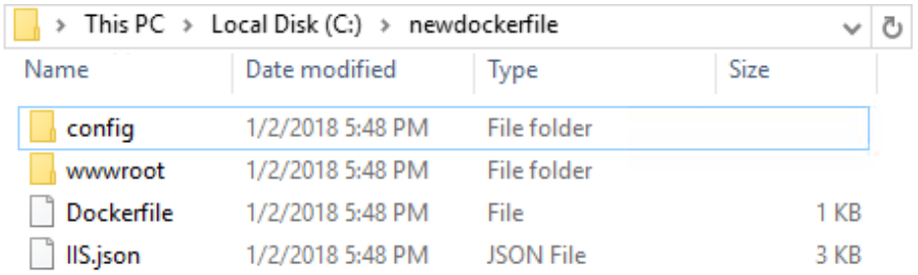


Generate Dockerfile with Image2Docker

Generate DockerFile with Image2Docker

ConvertTo-Dockerfile creates a folder which includes a dockerfile and all the files needed to generate the container

```
ConvertTo-Dockerfile `
    -OutputPath c:\newdockerfile `
    -Artifact IIS `
    -Local
```



dockerfile - Notepad

File Edit Format View Help

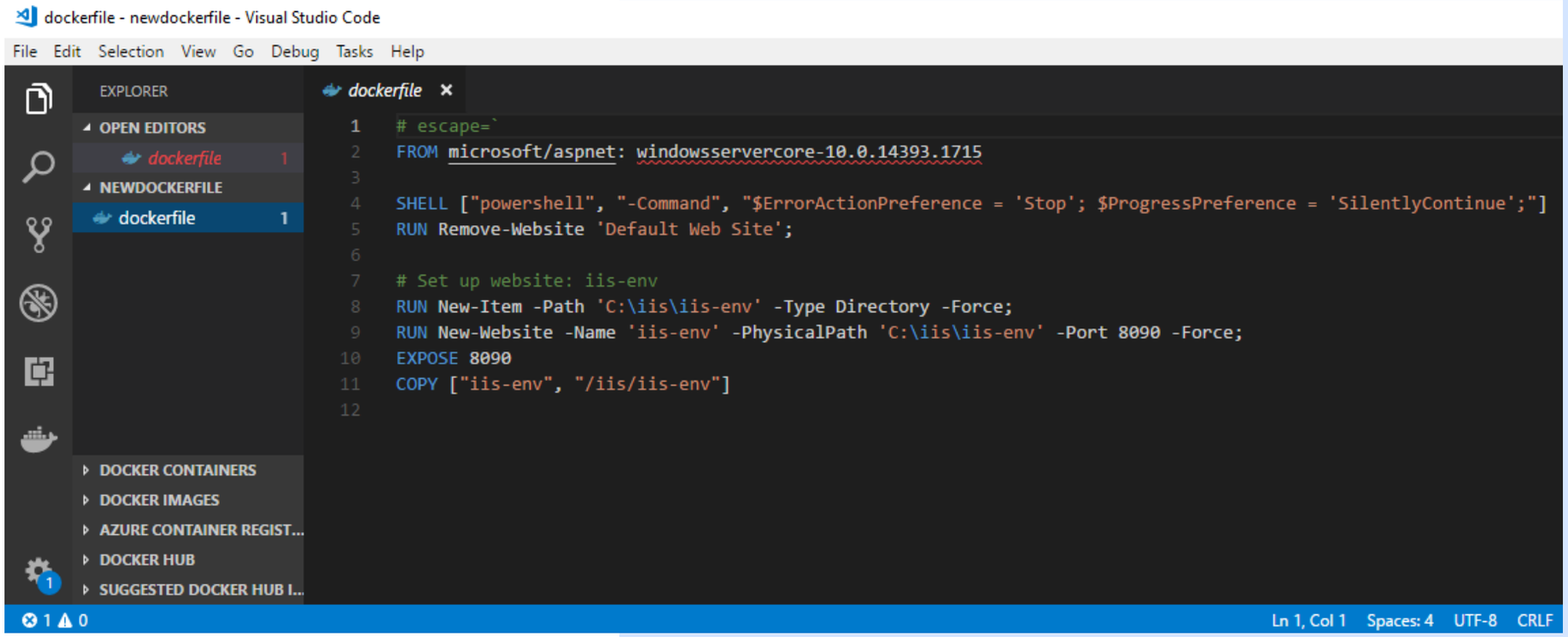
```
# escape=`
FROM microsoft/aspnet: windowsservercore-
10.0.14393.1715

SHELL ["powershell", "-Command",
"$ErrorActionPreference = 'Stop';
$ProgressPreference = 'SilentlyContinue';"]

RUN Remove-Website 'Default Web Site';

# Set up website: iis-env
RUN New-Item -Path 'C:\iis\iis-env' -Type
Directory -Force;

RUN New-Website -Name 'iis-env' -PhysicalPath
'C:\iis\iis-env' -Port 8090 -Force;

EXPOSE 8090

COPY ["iis-env", "/iis/iis-env"]
```

Tools – Visual Studio Code

Tools - Visual Studio Code

VS Code with the Docker Extension understands the structure of Dockerfiles as well as the available instructions

- Provides Intellisense type experience
- Free tool to aid in writing dockerfiles



Available as free download

Visual Studio: Visual Studio Code Download
Visual Studio: VS Code Docker Extensions

SQL Server 2016 Container Image

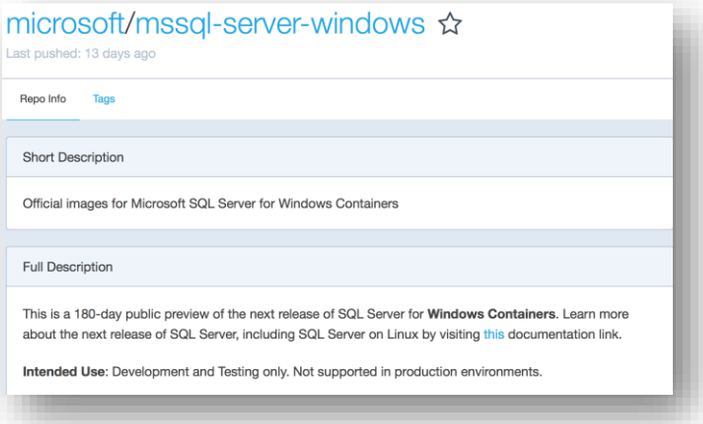
SQL Server in Docker

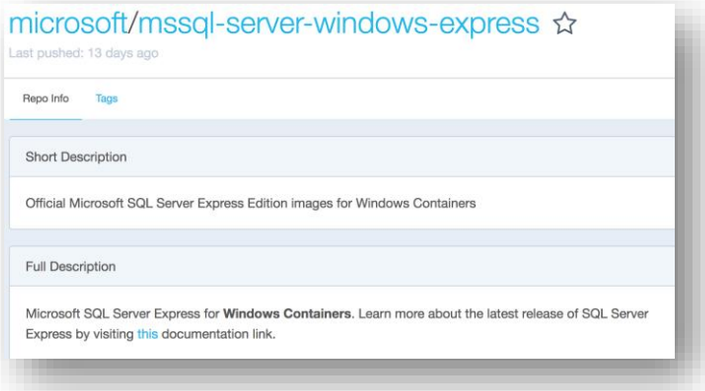
This GitHub repository aims to provide a centralized location for community engagement. In here you will find documentation, Dockerfiles and additional developer resources.

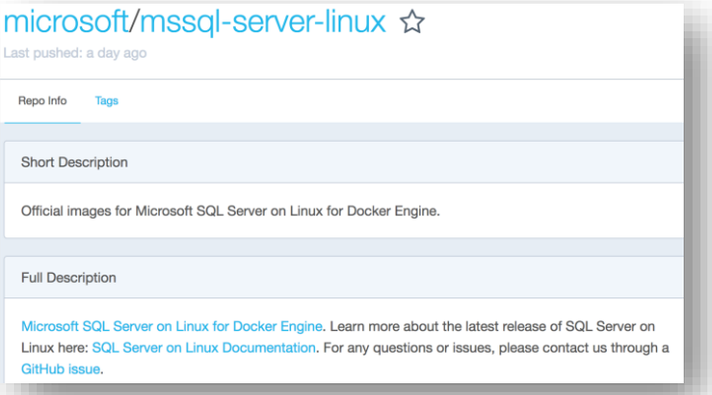
SQL Server in Docker comes in two different flavors:

- Linux-based containers: This Docker image uses SQL Server on Linux on top of an Ubuntu 16.04 base image. This is meant to be run on Docker Engine on its multiple platforms. There are also Dockerfiles here for building CentOS & RHEL based images.
- Windows-based containers: These Docker images use SQL Server 2016 SP1 Express Edition, SQL Server 2016 SP1 Developer Edition and SQL Server 2017 Evaluation Edition. All three images are based on Windows Container technology and can only be run using Docker Engine for Windows Containers.

Visit the Microsoft Docker Hub page for more information and additional images.







Microsoft

Getting Stated with Windows Containers

Overview of Container Features

Microsoft Services

Container Features

Feature Overview

In the following slides we'll go over the following Container features

- Identity Options
- Resource Controls
- Storage
- Named Pipe support
- Networking
- MSMQ
- Windows Admin Center (WAC)
- Containers in Azure
- Container Orchestration Options

Container Identity

Emulated Domain Join

- Containers cannot be domain-joined
- Containers do not store Passwords or certificate private keys
- By default, Services run under the Local System or Network Service of the Container Host's computer account
- Emulated Domain Join is used to pass a domain-based identity to the Container
- Allows portability between multiple AD environments, since the identity is passed to the container with docker run when it is started

How does it work

- The Container is configured to use a Group Management Service Account (gMSA) as the Container's identity
- Local System and Network Service on the Container Host are used as a proxy for the gMSA account
- Credentials are passed to the container at creation with docker run using the

  –security-opt "credentialspec=<*file*>"

Click for Demo ➔
(Works in Slide Show View only)

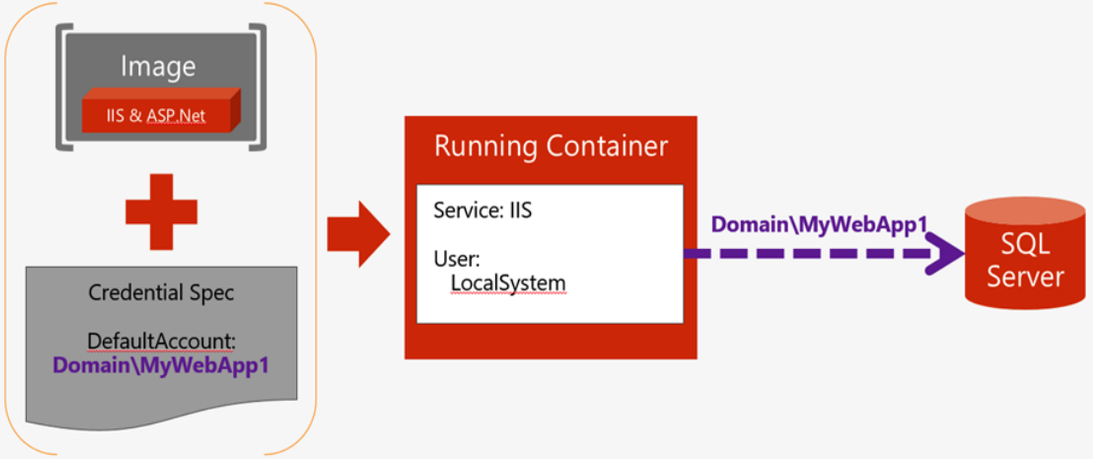
Windows Server 2016

Container Identity

Improved: Group Managed Service Accounts (gMSA)

Starting with Windows Server 2019, improvements in scalability and reliability of containers that use gMSA to access network resources have been made

- Fewer authentication errors when using a single gMSA with multiple containers instances
- No longer need to set the container's host name to be the same as the gMSA
- Fixed a bug that prevented using gMSA with Hyper-V isolated containers



Windows Server 2019

Resource Controls

Control Resources used by a Container

CPU (docker --cpu)

- Control CPU usage, quota, sharing weight, and period scheduler

Disk I/O (docker --device, --blkio)

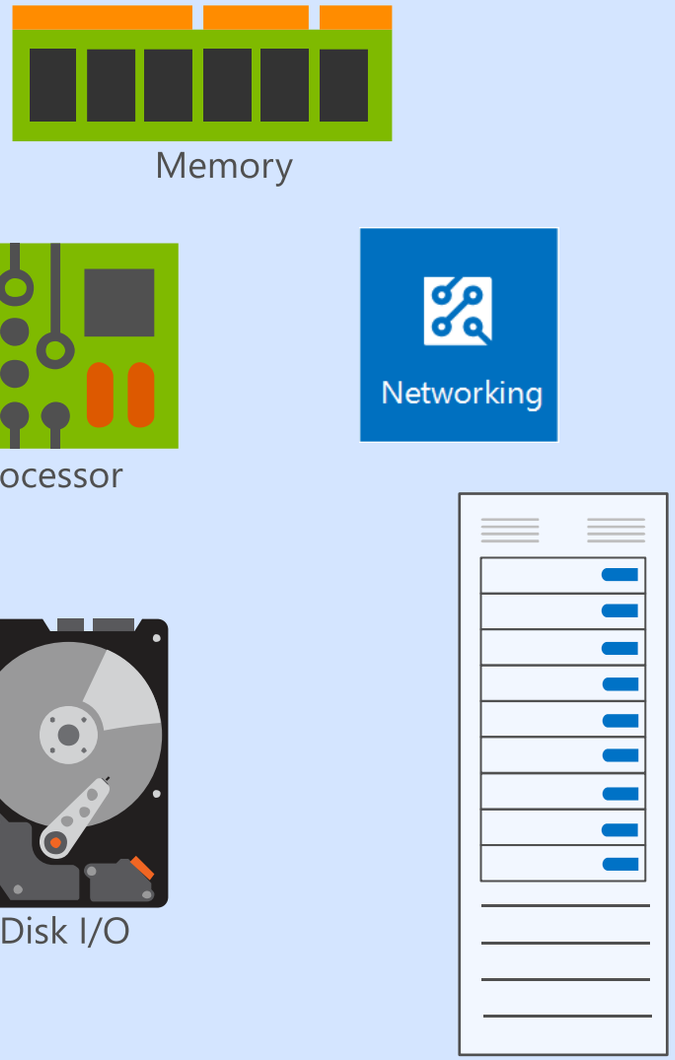
- Set bandwidth (bps) and Limit IO per second (iops)

Memory (docker --memory, --kernel-memory)

- Set memory limits

Network (Set-ContainerNetworkAdapter)

- Limit network bandwidth used by a Container



Storage – Persistent Volumes

Store persistent data for Container on local drives attached to the Container Host

Bind mount Example

- Map c:\cdata on Host to c:\data in Container

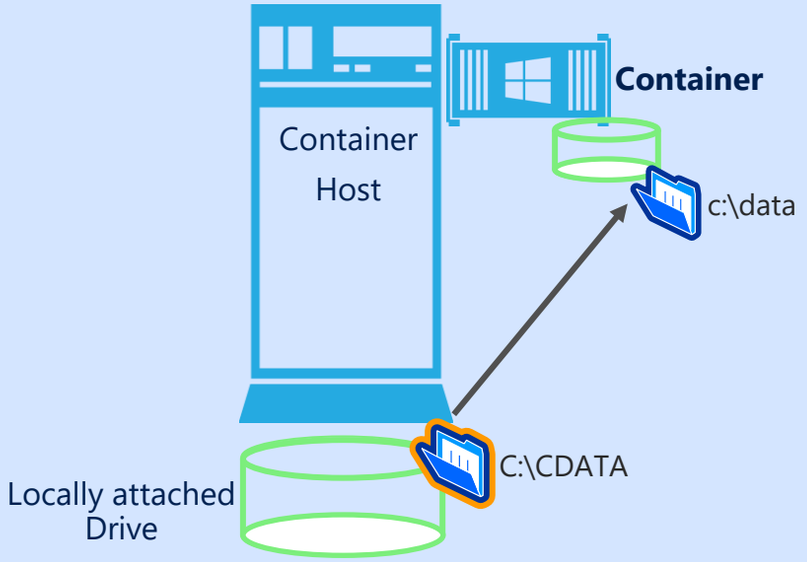
  Docker run –v c:\cdata:c:\data

Named volume Example

- Create a volume called “CDATA” on host and map to c:\data in container

  Docker volume create CDATA

  Docker run –v CDATA:c:\data



Storage – Clustered Shared Volumes

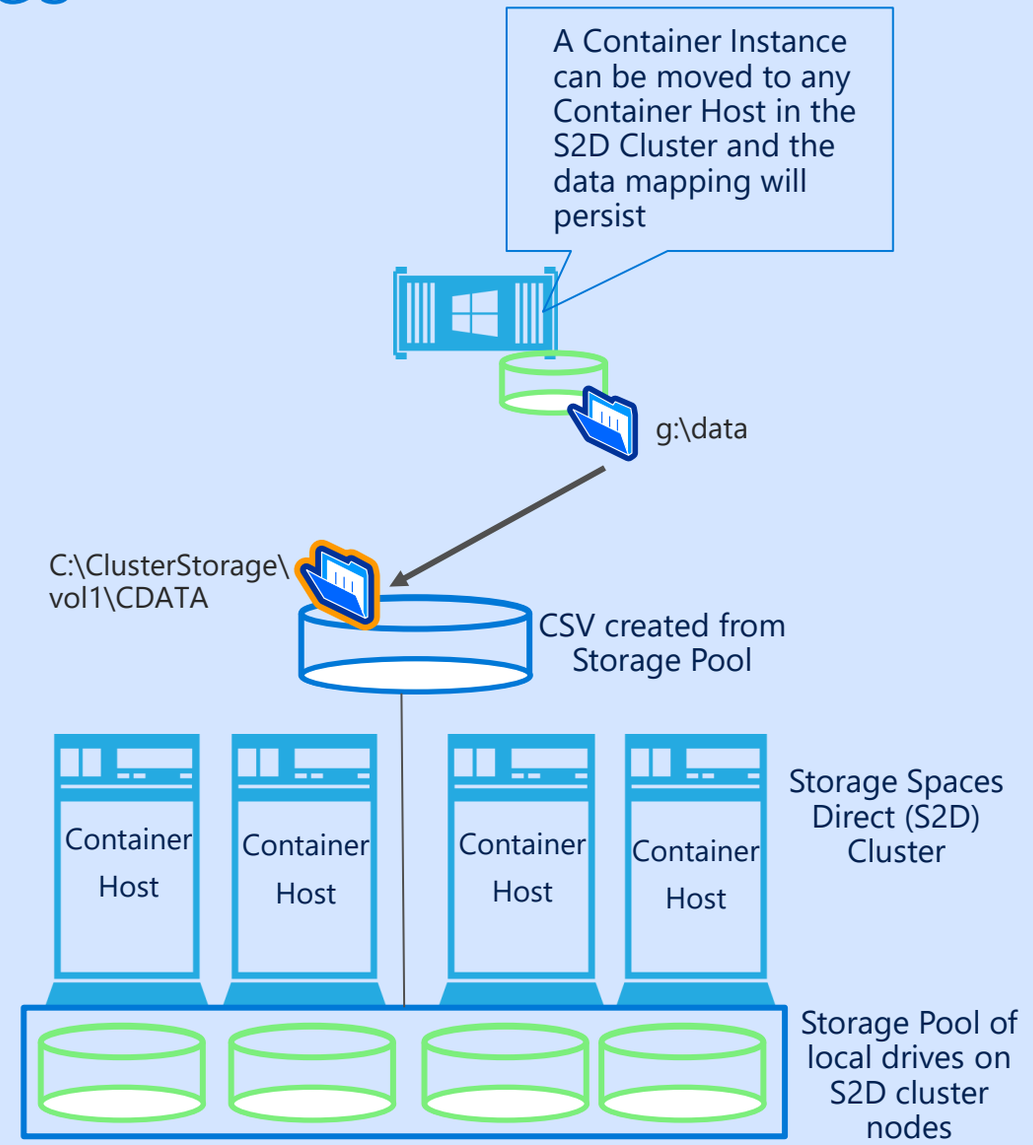
Introduced with Windows Server, version 1709, and backwards compatible with Windows Server 2016 with cumulative updates (Nov 2017)

Map Container data volumes to a Clustered Shared Volume (CSV) that is backed by

- Storage Spaces Direct (S2D) shared volume
- Traditional SAN (iSCSI, FCoE)

This allows containers to access persistent data no matter which node a Container Orchestrator places the container instance on

Docker run –name Demo –v c:\clusterstorage\vol1\cdata:g:\data



Storage – SMB Global Mapping

New Feature in Windows Server, version 1709

Map a data volume in a Container to a drive letter on the Container Host mapped to a remote SMB share with

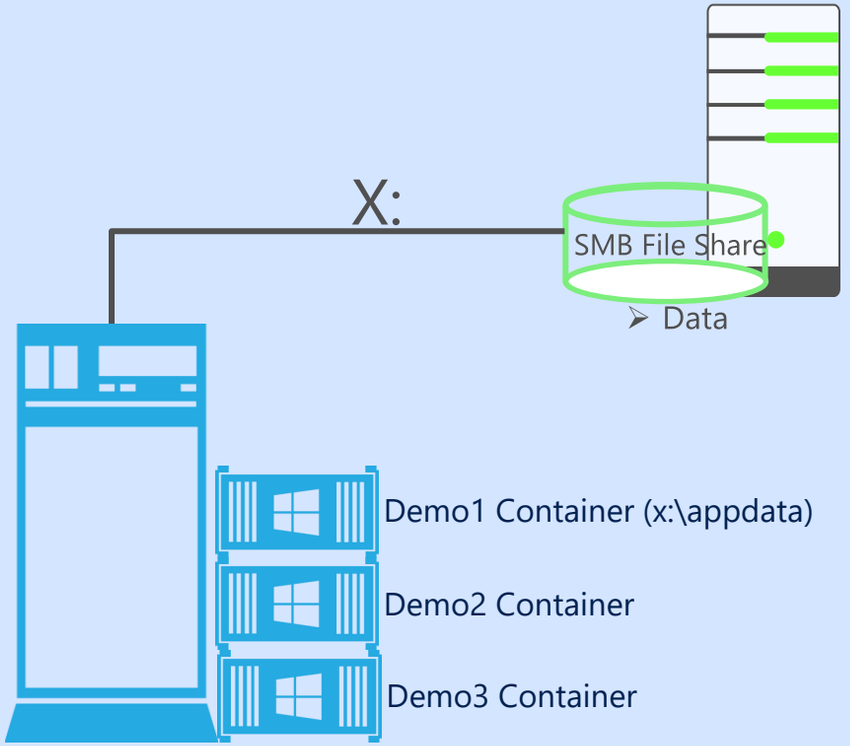
- New-SmbGlobalMapping
- Docker run –name <*container*> –v x:\data:x:\appdata

Cautions

- All users and apps on the host can also access the remote share
- Global mappings are not persistent. Must be reestablished by an orchestrator, agent or admin after server reboot
- Does not support DFS, DFSN or DFSR shares
- The target needs to be full SMB2 (or above)
- Read only global mappings not supported (yet!)

Example of mapping

```
$creds = Get-Credential
New-SmbGlobalMapping -RemotePath \\server\share1 -Credential $creds -LocalPath X:
Docker run –name demo1 –v x:\data:x:\appdata
```



Windows Server, version 1709

Container Networking

WinNAT and Custom networks

The Docker engine creates a default WinNAT network on the Container Host when it starts the first time

- Adapter: vEthernet (HNS Internal NIC)
- Network name: nat
- Internal vSwitch: DockerNAT

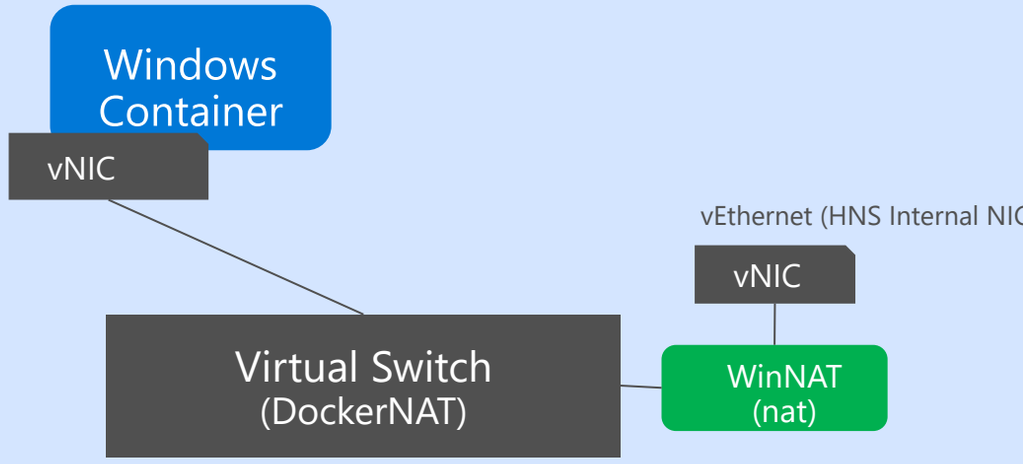
Unless specified, a container creates a virtual NIC connected to the **nat** network on startup

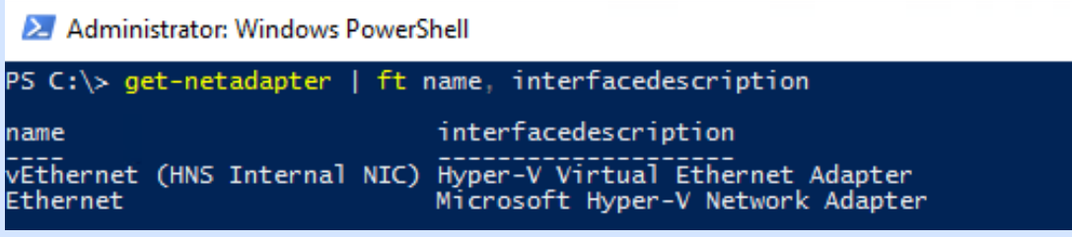
Custom networks are created using

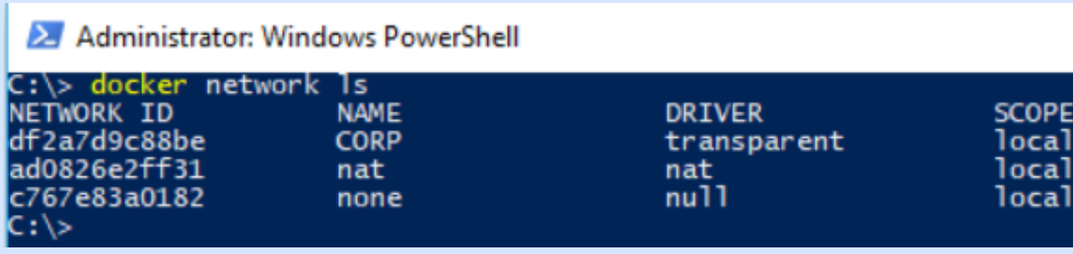
docker network create –d <network driver type> <Name>

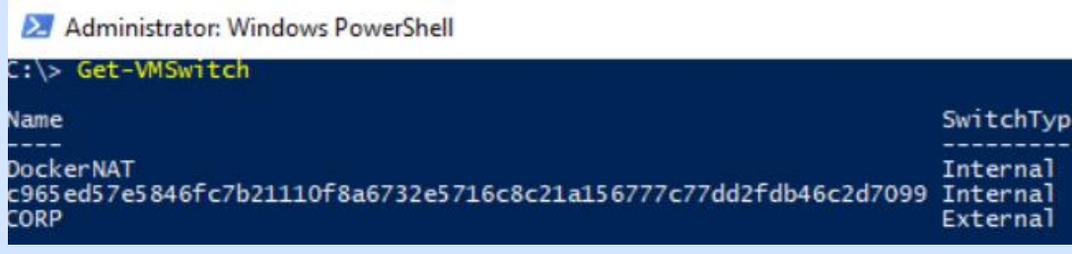
A Container is connected to a different network using

docker run --network=<network name>









Windows Server 2016

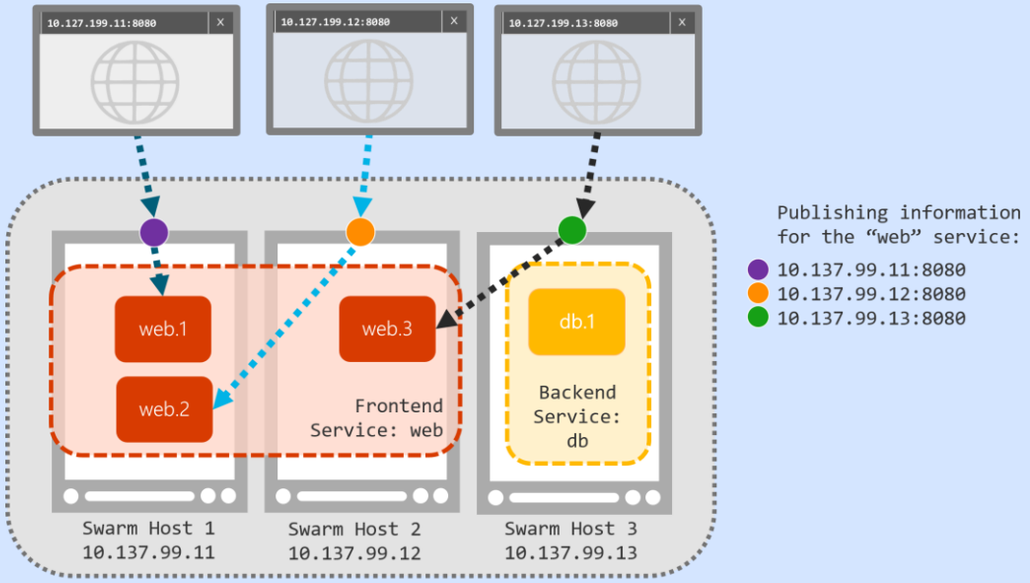
Container Networking – Docker Routing Mesh Support

New Feature in Windows Server, version 1709

- Docker's default option for defining published ports for Services when using Swarms
- Swarms enable deployment of containers to a cluster of Container Hosts
- Previously supported for Linux, but not for Windows

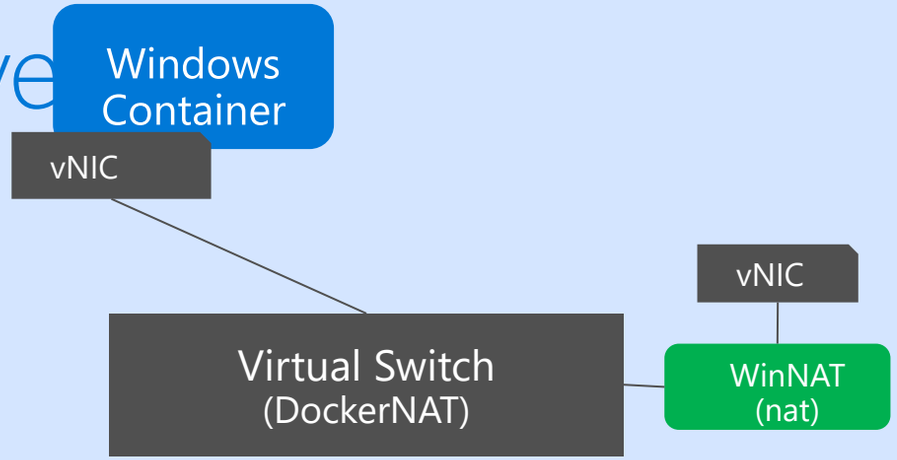
Example: Web service published to port 8080 of a Swarm made up of 3 container hosts

Docker service create –name web –replicas 3 –publish 8080:80 web-frontend



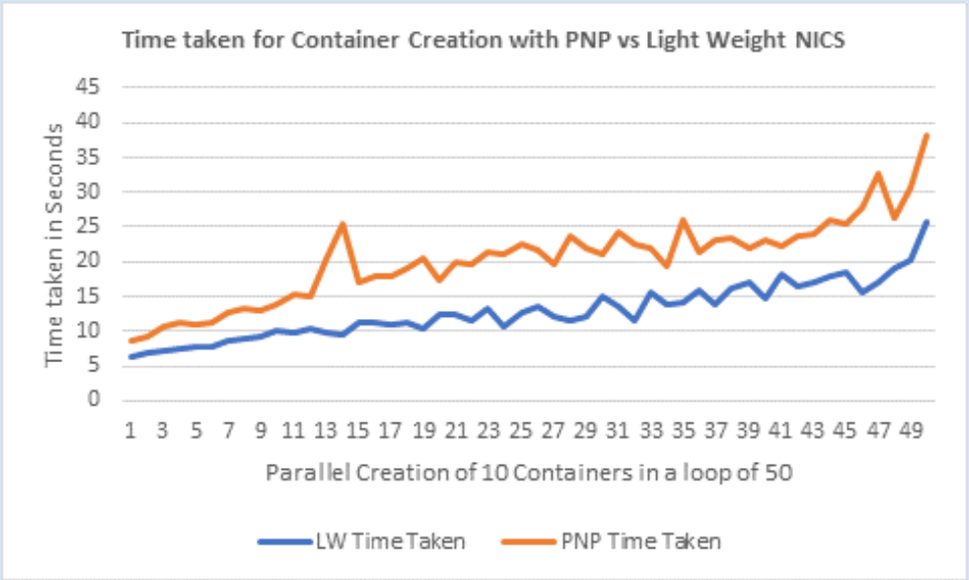
Windows Server, version 1709

Scalability and Startup Time Improve



Introduced in Windows Server, version 1803, Deviceless vNICs for Containers removes the overhead of using Windows PNP device management to make both endpoint creation and removal significantly faster.

- Network endpoint creation time in particular can have a notable impact on large-scale deployments, where scaling up and down can add unwanted delay
- Increased scalability of Windows Server Containers from 50 to 500 containers on one host with linear network endpoint creation cost
- Decreased Windows Server Container start-up time with 30% improvement in network endpoint creation time and 50% improvement in time taken for container deletion



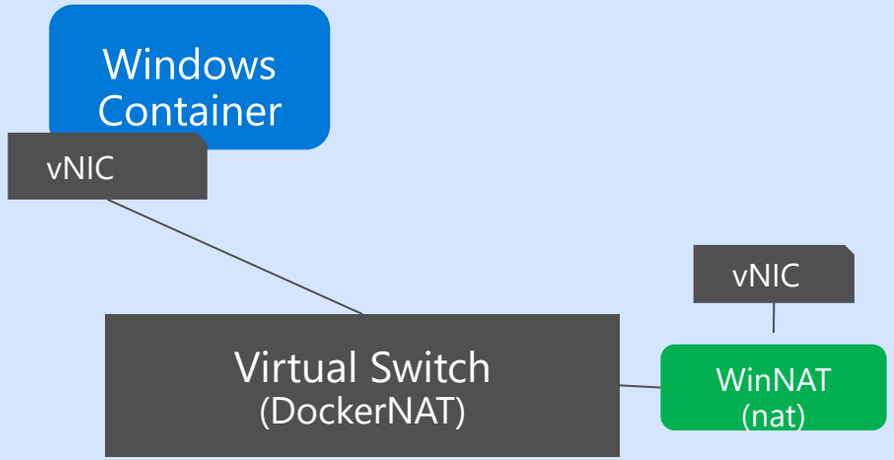
Windows Server, version 1803

TechNet Blog: Improvements to deviceless vNICs

NAT Performance Improvements added in Windows Server, version 1803

Introduced in Windows Server, version 1803 based on customer feedback, built-in NAT optimizations added

- Optimizations (CPU utilization) of machinery used for translation decisions of incoming traffic
- Widened network throughput pipeline by 10-20%
- This is needed for Windows use-cases, such as Windows Defender Application Guard in the Microsoft Edge web browser or Docker for Windows which rely heavily on network address translation (NAT)



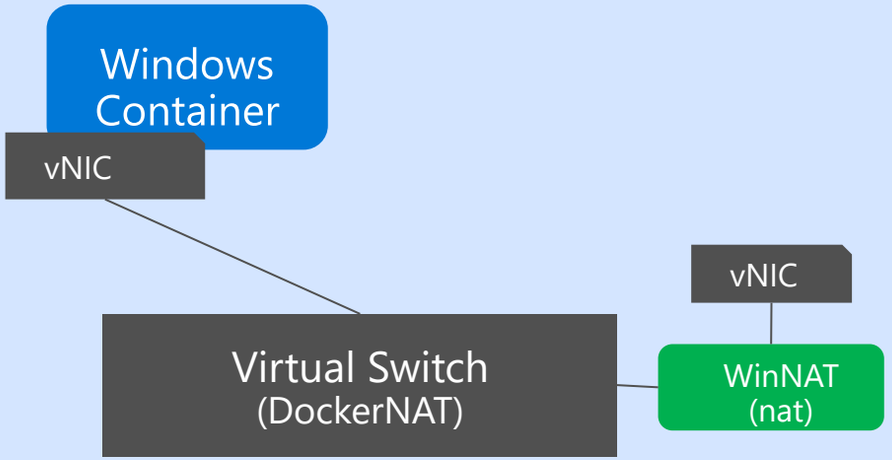
Windows Server, version 1803

TechNet Blog: Improvements to deviceless vNICs

Network Quality Improved in Windows Server, version 1803

Enhancements to provide a healthy, consistent and sustainable networking experience of the container ecosystem on Windows

- Stabilized DNS resolution within containers out-of-the-box
- Enhanced stability of Kubernetes services on Windows
- Improved recovery after Kubernetes container crashes
- Fixes to address and port range reservations through WinNAT
- Improved persistence of containers after Host Networking Service (HNS) restart
- Improved persistence of containers after unexpected container host reboot
- Better overall resiliency of NAT networking



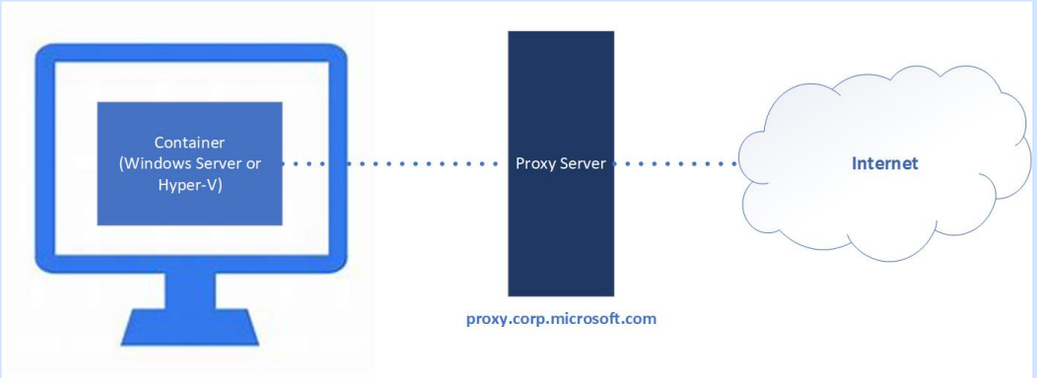
Windows Server, version 1803

TechNet Blog: Container Network Improvements

Http Proxy Support added in Windows Server, version 1803

A container host can inject proxy settings upon container instantiation, such that container traffic is forced through the specified proxy

- Supported on both Windows Server and Hyper-V Containers
- Gives developers more control and flexibility over their desired container network setup



Localhost/loopback support add in Windows Server, version 1803

Access containerized web services via “localhost” or 127.0.0.1 (loopback)

Windows Server, version 1803

Docker Docs: Container Proxy Support
TechNet Blogs: Access to container ports via localhost

Host Device Access for Containers added in Windows Server 2019

Added the capability to assign simple buses to process-isolated Windows Server containers

- Applications running in containers that need to talk over SPI, I2C, GPIO, and UART/COM will now be able to do so

Windows Server 2019

TechNet Blog: Bringing Device Support to Containers

Named Pipes

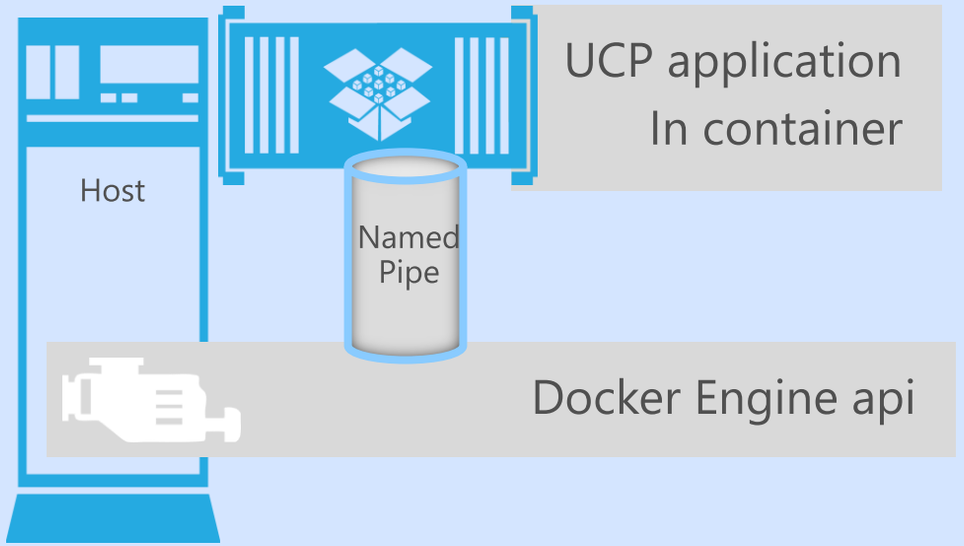
New: Named Pipes support

Bind mountable APIs on a container host can now be accessed from Windows Containers

Example of use

- The Docker API is bind mountable
- Software, like Docker Universal Control Plane (UCP) can be packaged in a container.
- UCP can access the Docker API on the container host to start more containers or visualize containers, network and volumes on a container host

docker run –v \\.\pipe\docker_engine:\\.\pipe\docker_engine



Windows Server, version 1709

MSMQ

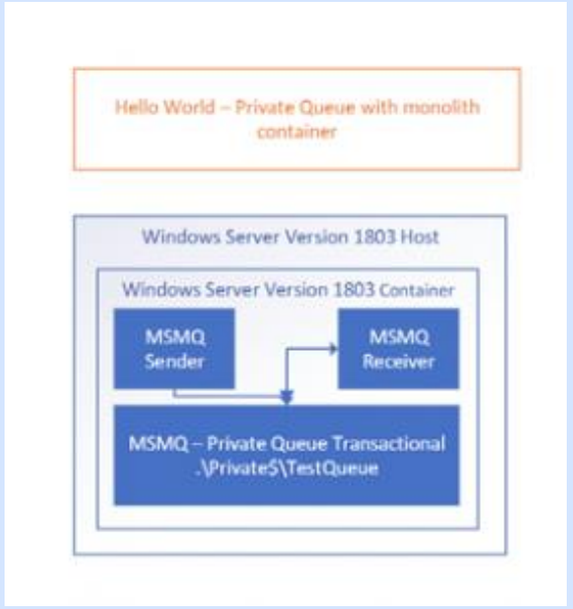
MSMQ Support added in Windows Server, version 1803

MSMQ can be installed in a Windows Container

- This is one of the top asks from customers who are lifting and shifting traditional apps with containers

  https://windowsserver.uservoice.com/forums/304624-containers/suggestions/15719031-create-base-container-image-with-msmq-server

- Requires Windows Server Core base image for Windows Server, version 1803 (or later)



Windows Server, version 1803

TechNet Blog: Hello World MSMQ! ...from Containers

Knowledge Check

1. What are the two types of Windows containers and how do they differ?
2. What container runtime is required to be installed in order to work with Windows containers?
3. What are the three Winodws container base images?