

Docker on Windows Server 2016

Friday, August 4, 2017 2:26 PM

Install and configure Docker, along with deploying and managing Windows-based containers, on a Windows Server 2016 server.

This is a short workshop to introduce you to Windows-based containers. In this workshop, you will gain experience in installing and configuring Docker on a Windows 2016 Server server. You'll then deploy a couple of different images as containers to the server and experiment with managing those images and containers. Finally, you will configure Azure to allow you to access those containers from outside of your virtual network.

What You Will Learn

- Installing and Configuring Docker on Windows Server 2016
- Downloading and Managing Images
- Deploying and Working With Containers
- Exposing Docker Services in Azure

Ideal Audience

- IT Managers
- Developers and Software Architects
- Configuration and Change Managers
- DevOps Engineers

Overview

This is a short workshop to introduce you to Windows-based containers. In this workshop, you will gain experience in installing and configuring Docker on a Windows Server 2016 server. You'll then deploy a couple of different images as containers to the server and experiment with managing those images and containers. Finally, you will configure Azure to allow you to access those containers from outside of your virtual network.

Time Estimate: 2.5 hours

Requirements

Setup Requirements

The following workshop will require that you use a Remote Desktop client in order to connect to a remote machine. If you are using a Mac, then [download](#) the Microsoft Remote Desktop client.

Additional Requirements

For the following workshop, you will need a subscription (trial or paid) to Microsoft Azure. Please see the [next](#) page for how to create a trial subscription, if necessary.

Azure Registration

Azure

We need an active Azure subscription in order to perform this workshop. There are a few ways to accomplish this. If you already have an active Azure subscription, you can skip the remainder of this page. Otherwise, you'll either need to use an Azure Pass or create a trial account. The instructions for both are below.

Azure Pass

If you've been provided with a voucher, formally known as an Azure Pass, then you can use that to create a subscription. In order to use the Azure Pass, direct your browser to <https://www.microsoftazurepass.com> and, following the prompts, use the code provided to create your subscription.

Trial Subscription

Direct your browser to <https://azure.microsoft.com/en-us/free/> and begin by clicking on the green button that reads **Start free**.

1. In the first section, complete the form in its entirety. Make sure you use your *real* email address for the important notifications.
2. In the second section, enter a *real* mobile phone number to receive a text verification number. Click send message and re-type the received code.
3. Enter a valid credit card number. **NOTE:** You will *not* be charged. This is for verification of identity only in order to comply with federal regulations. Your account statement may see a temporary hold of \$1.00 from Microsoft, but, again, this is for verification only and will "fall off" your account within 2-3 banking days.
4. Agree to Microsoft's Terms and Conditions and click **Sign Up**.

This may take a minute or two, but you should see a welcome screen informing you that your subscription is ready. Like the Office 365 trial above, the Azure subscription is good for up to \$200 of resources for 30 days. After 30 days, your subscription (and resources) will be suspended unless you convert your trial subscription to a paid one. And, should you choose to do so, you can elect to use a different credit card than the one you just entered.

Congratulations! You've now created an Office 365 tenant; an Azure tenant and subscription; and, have linked the two together.



Exploring Azure

Objective

The first objective is for you to become familiar with connecting to and navigating the Azure portal. This will not be a difficult exercise, but will nonetheless demonstrate how to work within the Azure user interface.

Azure Portal Basics

Let's start by connecting to the Azure portal and becoming familiar with navigation.

1. Open a browser and navigate to <http://www.azure.com>.
2. In the top-right corner of your screen, you will see the menu option **PORTAL**. Click on it.
3. If you have not already, you will be required to authenticate.
4. After authentication is successful, you will be directed to your *Dashboard*. The dashboard is configurable by adding, removing and resizing *tiles*. Additionally, you can have multiple dashboards depending on your preferences. You could have different dashboards for resources dedicated to different functions, lines of business, or for operations.
5. On the left will be your primary navigational menu. You should see a list of favorited services on the menu with descriptions. (NOTE: The number of options listed in your menu may differ from that of others depending on the number of services you have selected as a favorite.) If all you see are icons (no descriptions) on your menu, your menu is currently collapsed. Click the "hamburger"  to expand it.
6. Pretty close to the top of your menu, you should see **Resource Groups** . Click this option.
7. Upon clicking the Resource Groups menu item, a *blade* will open revealing any created resource groups. In order to create resources in Azure, you must assign/place it in a resource group.

This is where we will get started creating our resources.

While this introduction wasn't too technical, it is sufficient for getting us to a point where we can begin the specifics in the workshop. If you'd like to look around a bit more, click a few of the other options in the main menu. Then, when you are ready, can you proceed to the [next](#) step.




Create a Virtual Machine

Objective

Now that we've explored the Azure portal a bit, let's get started with creating some resources. Our primary resource will be a virtual machine on which we install Docker. Once we create the virtual machine, we'll see that some additional resources are created for us.

Create a Resource Group


As stated on the previous page, in order to create resources, we need a *Resource Group* to place them in.

1. If you are not there already, go ahead and click on the **Resource Groups**  in the Azure Portal to open the Resource Groups blade.
2. At the top of the Resource Groups blade, click on **Add** . This will open a panel that asks for some basic configuration settings.
3. Complete the configuration settings with the following:
 - Resource group name: **azworkshops_docker_win_demo**
 - Subscription: **<choose your subscription>**
 - Resource group location: **<choose your location>**
4. *<Optional>* Check *Pin to dashboard* at the bottom of the panel.
5. Click **Create**.
6. It should only take a second for the resource group to be created. Once you click create, the configuration panel closes and returns you to the list of available resource groups. Your recently created group may not be visible in the list. Clicking on **Refresh**  at the top of the Resource Groups blade should display your new resource group.

NOTE: When you create a resource group, you are prompted to choose a location. Additionally, as you create individual resources, you will also be prompted to choose locations. The location of resource groups and their resources can be different. This is because resource groups store *metadata* describing their contained resources; and, due to some types of compliance that your company may adhere to, you may need to store that metadata in a different location than the resources themselves. For example, if you are a US-based company, you may choose to keep the metadata state-side while creating resources in foreign regions to reduce latency for the end-user.













Create a Virtual Machine

Now that we have an available resource group, let's create the actual Windows server.

1. If you are not there already, go ahead and navigate to the **azworkshops_docker_win_demo** resource group.
2. At the top of the blade for our group, click on **Add** . This will display the blade for the *Azure Marketplace* allowing you to deploy a number of different solutions.
3. We are interested in deploying a Windows Server 2016 Datacenter server. Therefore, in the *Search Everything* box, type in **Windows Server 2016**. This will display a couple of different versions. Choose **Windows Server 2016 Datacenter**.



4. There will be a number of solutions available, including one with containers already enabled. For the practice, we'll enable containers manually. Therefore, choose the image as highlighted in the image below.

Windows Server 2016 Datacenter			
Results			
NAME	PUBLISHER	CATEGORY	
 Windows Server 2016 Datacenter	Microsoft	Compute	
 [HUB] Windows Server 2016 Datacenter	Microsoft	Compute	
 Windows Server 2016 Datacenter	Microsoft		
 Windows Server 2016 Datacenter - with Containers	Microsoft	Compute	
 [smalldisk] Windows Server 2016 Datacenter	Microsoft	Compute	
 Windows Server 2016 Datacenter - Server Core	Microsoft	Compute	
 [smalldisk] Windows Server 2016 Datacenter - Server Core	Microsoft	Compute	
 SharePoint Server 2016 Trial	Microsoft	Compute	
 HPC Pack 2016 Head Node on Windows Server 2016	Microsoft	Compute	
 HPC Pack 2016 Compute Node on Windows Server 2016	Microsoft	Compute	
 HPC Pack 2016 Head Node on Windows Server 2012 R2	Microsoft	Compute	
 HPC Pack 2016 Compute Node on Windows Server 2012 R2	Microsoft	Compute	

5. This will display a blade providing more information about the server we have chosen. To continue creating the server, choose **Create**.

6. We are now prompted with some configuration options. There are 3 sections we need to complete and the last section is a summary of our chosen options.

1. Basics

- Name: **docker-win**
- VM disk type: **SSD**
- Username: **localadmin**
- Password: **Pass@word1234**
- Confirm password: *<same as above>*
- Subscription: *<choose your subscription>*
- Resource group: **Use existing - azworkshops_docker_win_demo**
- Location: *<choose a location>*
- Already have a Windows Server license? **No**

2. Size

- **DS1_V2**

3. Settings

- Use managed disks: **No**

- Storage account: (click on it & **Create New**)
 - Name: **dockerwindata**<random number> (ex. *dockerwindata123456*)
(NOTE: This name must be *globally* unique, so it cannot already be used.)
 - Performance: **Premium**
 - Replication: **Locally-redundant storage (LRS)**
 - Virtual network: <accept default> (e.g. *(new) azworkshops_docker_win_demo-vnet*)
 - Subnet: <accept default> (e.g. *default (172.16.1.0/24)*)
 - Public IP address: <accept default> (e.g. *(new) docker-win-ip*)
 - Network security group (firewall): <accept default> (e.g. *(new) docker-win-nsg*)
 - Extensions: **No extensions**
 - Availability set: **None**
 - Boot diagnostics: **Enabled**
 - Guest OS diagnostics: **Disabled**
 - Diagnostics storage account: (click on it & **Create New**)
 - Name: **dockerwindiags**<random number> (ex. *dockerwindiags123456*)
 - Performance: **Standard**
 - Replication: **Locally-redundant storage (LRS)**
4. Summary (just click **OK** to continue)

Once scheduled, it may take a minute or two for the machine to be created by Azure. Once it has been created, Azure *should* open the machine's status blade automatically.

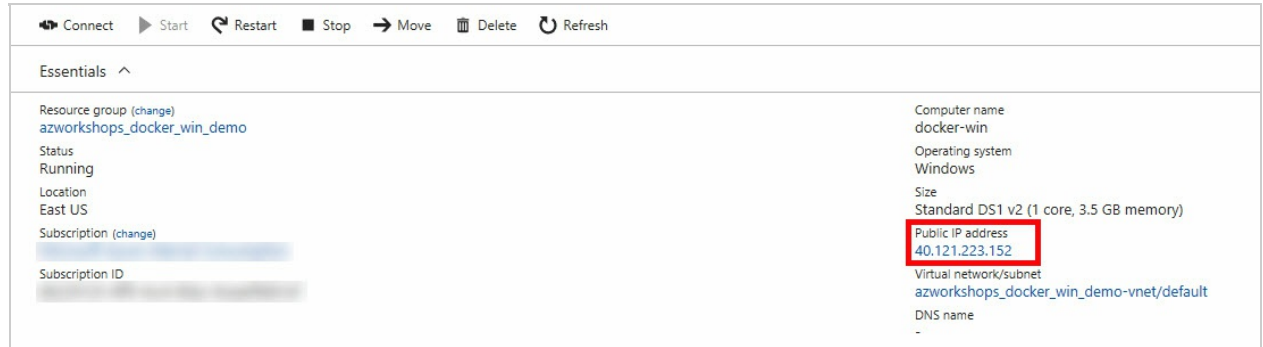
Connect to the Virtual Machine

Once your machine has been created, we can remotely connect to it via a remote desktop protocol (RDP) client.

Get Public IP

1. If it is not already open, navigate to the **Overview** blade of your newly created virtual machine.



2. In the top section of the blade, in the right column, you should see a **Public IP address** listed.



3. Copy the IP address.

Connect to the Machine via Remote Desktop

To connect to the machine remotely, we need to download the Remote Desktop Protocol (RDP) profile.

1. Click on the **Overview**  to return to the general information for the **ad-connect** virtual machine.
2. In the **Actions** section, click on **Connect** . This will download the RDP profile to your machine.
3. Open the profile and accept any warnings.
4. For the username, enter **\localadmin** (with the backslash). And, for the password, enter **Pass@word1234**. Click **OK**.
5. Again, accept any warnings.

Congratulations. You have successfully created and connected to your remote Windows Server 2016 server in Azure. You are now ready to install the Docker runtime.

Install Docker

Overview

We have just created our Windows Server 2016 server. We now need to apply any available system updates along with installing and configuring Docker to begin working with containers.

Install Updates

Just like any other operating system, updates are periodically released to support new features and patch any potential security threats. We will apply the updates first.

1. If you have not already, connect to your remote Windows Server 2016 server and login.
2. Open a command prompt as an Administrator, type the following at the command prompt:

```
sconfig
```

3. This will open a screen like the following:

```
Administrator: Command Prompt - sconfig
Microsoft (R) Windows Script Host Version 5.812
Copyright (C) Microsoft Corporation. All rights reserved.

Inspecting system...

=====
                        Server Configuration
=====

1) Domain/Workgroup:           Workgroup:  WORKGROUP
2) Computer Name:             DOCKER-WIN
3) Add Local Administrator
4) Configure Remote Management  Enabled
5) Windows Update Settings:    DownloadOnly
6) Download and Install Updates
7) Remote Desktop:            Enabled (more secure clients only)
8) Network Settings
9) Date and Time
10) Telemetry settings         Enhanced
11) Windows Activation
12) Log Off User
13) Restart Server
14) Shut Down Server
15) Exit to Command Line

Enter number to select an option: 6
```

4. Choose option 6 , then A (twice) to download and install all updates.
5. Depending on the number and size of available updates, this process may take a few minutes and could require a reboot. Now would be a good time to take a break.

Install Docker

We now have an updated Windows operating system. We are ready to install Docker.

1. Open a PowerShell prompt as an Administrator and type the following:

```
Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201 -Force
Install-Module -Name DockerMsftProvider -Force
Install-Package -Name docker -ProviderName DockerMsftProvider -Force
Restart-Computer -Force
```

2. This will download the Docker engine and install it as a background service.
3. After you run the above commands, your virtual machine will reboot forcing a disconnect. Go ahead and reconnect.

Ensure Docker Engine is Running

1. Open a PowerShell prompt as an Administrator and type the following:

```
docker version
```

2. You should see something similar to the following:

```
Client:
  Version:      17.03.1-ee-3
  API version:  1.27
  Go version:   go1.7.5
  Git commit:   3fcee33
  Built:        Thu Mar 30 19:31:22 2017
  OS/Arch:     windows/amd64

Server:
  Version:      17.03.1-ee-3
  API version:  1.27 (minimum version 1.24)
  Go version:   go1.7.5
  Git commit:   3fcee33
  Built:        Thu Mar 30 19:31:22 2017
  OS/Arch:     windows/amd64
  Experimental: false
```

3. Because the service is running, we can now use the `docker` command later in this workshop.

You've successfully installed the Docker engine.

Nano Man

Overview

Now that we have Docker installed, we are able to deploy images as containers. In this short step of the workshop, we will deploy a small containers to test our Docker engine on Windows.

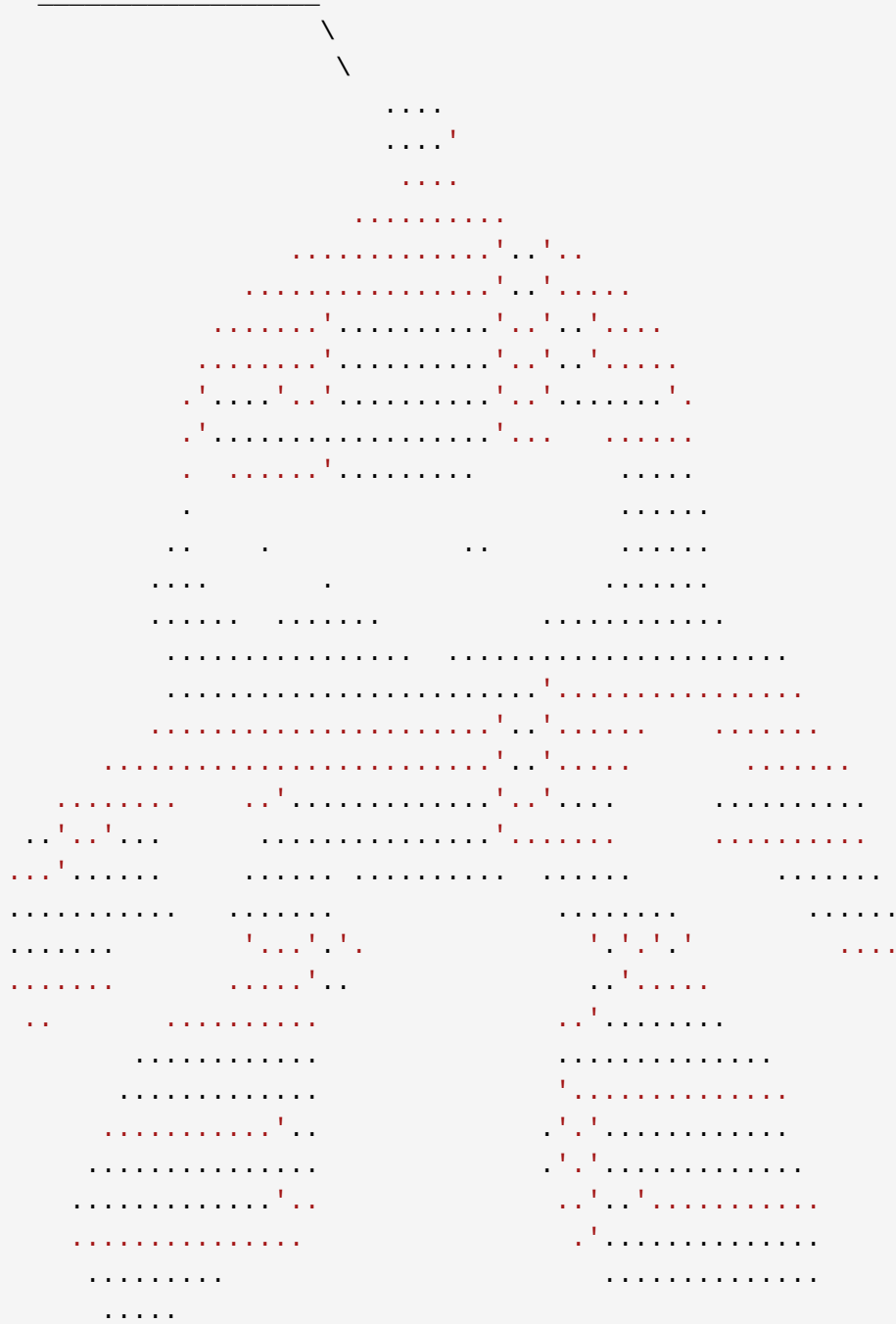
Hello World

1. Ensure you have logged in to your remote Windows Server 2016 server and are at the prompt.
2. From the PowerShell prompt, type the following:

```
docker run microsoft/dotnet-samples:dotnetapp-nanoserver
```

3. You should then see something similar to the following:

Dotnet-bot: Welcome to using .NET Core!



****Environment****

Platform: .NET Core 1.0

OS: Microsoft Windows 10.0.14393

Deploying an IIS Container

Overview

The final part of this workshop is to expose an IIS container outside of Azure. We're going to create a simple web server and access it from our local machine.

IIS

We are going to deploy a basic container hosting IIS and then expose the sample IIS website outside of Azure.

From PowerShell, type the following:

```
docker run -d -p 80:80 microsoft/iis
```

This will download and run IIS in the background. As stated earlier in this workshop, we often run services in *detached* mode (`-d`). As new parameter that you see here is mapping, or publishing (`-p`), ports - very similar to a NAT, if you are familiar with the concept. There are two ports specified here separated by a colon. The first number is the host's port while the second number is the container's port. So, in essence, we are mapping the host's port 80 to the container's port 80. If our container runs multiple services or a service requiring multiple ports, we can also specify a port range.

1. Let's make sure that IIS is running successfully. Open a web browser on the virtual server and try to navigate to `http://localhost` . Oops. It seems we received an error. What did we do wrong? Let's investigate.
2. If one is not already open, open a PowerShell window and type in `docker ps` .
3. You should see something like the following:

CONTAINER ID	IMAGE	COMMAND	CREATED
c21c24a24027	microsoft/iis	"C:\\ServiceMonitor..."	23 seconds ago
o	Up 14 seconds	0.0.0.0:80->80/tcp	kickass_raman

4. Notice the *Ports* column. Our external port is not mapped to the loopback address (e.g. 127.0.0.1 or localhost). Long story short, this is due to a way Windows maps its network interfaces.
5. We need to get the actual, virtual IP address of the container. To do this, type the following at the PowerShell prompt (change the container id to your container's id):

```
docker inspect --format '{{ .NetworkSettings.Networks.nat.IPAddress }}' c21
```

7. So, let's use the returned IP instead of the localhost to load our website. In the browser change the URL to http://<your container's virtual IP address>:8080 (e.g. http://192.168.150.195).

Windows Firewall


Before we can access the IIS server from outside of Azure, we need to open the port in Windows Firewall.




1. On the remote server, click the **Start Menu** and begin typing **Firewall**. This should provide a menu option for **Windows Firewall with Advanced Security**. Click on it.
2. Select **Inbound Rules** in the left pane and click **New Rule...** in the right pane.
3. For **Rule Type**, select **Port** and click **Next**.
4. On the **Protocol and Ports** page, select **TCP, Specific local ports**, and enter **80** in the input box. Click **Next**.
5. For **Action**, choose **Allow the connection** and click **Next**.
6. For **Profile**, leave all three profiles checked and click **Next**.
7. Finally, name the rule **Allow-HTTP** and click **Finish**.

Network Security Group (NSG)

Now that our web server is running, let's make it available outside of Azure.

When we created our Windows virtual machine, we accepted the defaults, including the default settings for our NSG. The default settings only allowed RDP (port 3389) access. We need to add a rule to our NSG to allow HTTP traffic over port 80.

1. If you are not still there, go back to the Azure portal and navigate to the settings of your CentOS virtual machine.
2. In the left menu, click on **Network interfaces**  .
3. This will open the *Network Interfaces* blade for your CentOS virtual machine. Click on the singular, listed interface.

4. In the left menu, click on **Network security group**  .
5. This will list the currently active NSG. In our case, it should be the NSG that was created with our virtual machine - **docker-centos-nsg**. Click on the NSG (**NOTE**: Click on the actual NSG link, **NOT** on **Edit**).
6. In the left menu, click on **Inbound security roles**  .
7. At the top of the blade, click **Add**  .
8. Enter the following configuration:
 - Name: **allow-http**
 - Priority: **1010**
 - Source: **Any**
 - Service: **HTTP**
 - Action: **Allow**
9. Click **OK**.

This should only take a couple of seconds. Once you see the rule added, open a new browser and navigate to the IP address of your Windows virtual machine, including the port number. The IP address used in this workshop's screen shots is **40.121.223.152** (your IP address will be different). Using the aforementioned IP address, I would direct my browser to **http://40.121.223.152/**. Doing so, you should see the IIS landing page.