

Microsoft Cloud Workshop

Containers and DevOps Hackathon
Pre-setup

November 2017

© 2017 Microsoft Corporation. All rights reserved. This document is confidential and proprietary to Microsoft. Internal use only. This document is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY.

This document is provided "as-is." Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

Some examples are for illustration only and are fictitious. No real association is intended or inferred.

Exercise 0: Before the Hackathon	1
Task 1: Resource Group	1
Task 2: Install Git tools	2
Task 3: Create an SSH key	7
Task 4: Create a build agent VM	8
Task 5: Connect securely to the build agent	11
Task 6: Complete the build agent setup	13
Task 7: Create an Azure Container Registry	15
Task 8: Create a Storage Account	16
Task 9: Create an Azure Container Service cluster	19

Exercise 0: Before the Hackathon

Duration: 30 minutes

You should follow all of the steps provided in Exercise 0 *before* attending the hackathon.

NOTE: All resources created for this hackathon should be assigned to a location where all Azure Container Registry SKUs are available. At the present time, these include East US, West Central US, and West Europe.

Task 1: Resource Group

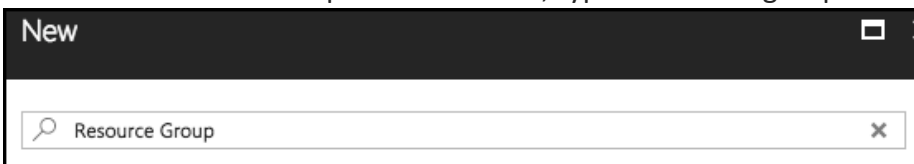
You will create an Azure Resource Group to hold most of the resources that you create in this hackathon. This approach will make it easier to clean up later. You will be instructed to create new resources in this Resource Group during the remaining exercises.

Tasks to complete

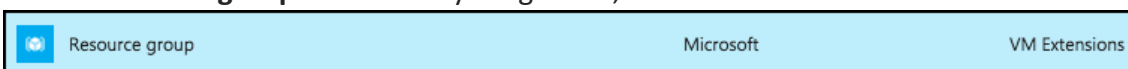
1. In your browser, navigate to the **Azure Portal** (<https://portal.azure.com>).
2. Select **+ New** in the navigation bar at the left.



3. In the Search the Marketplace search box, type "Resource group" and press Enter.



4. Select **Resource group** on the Everything blade, and select **Create**.

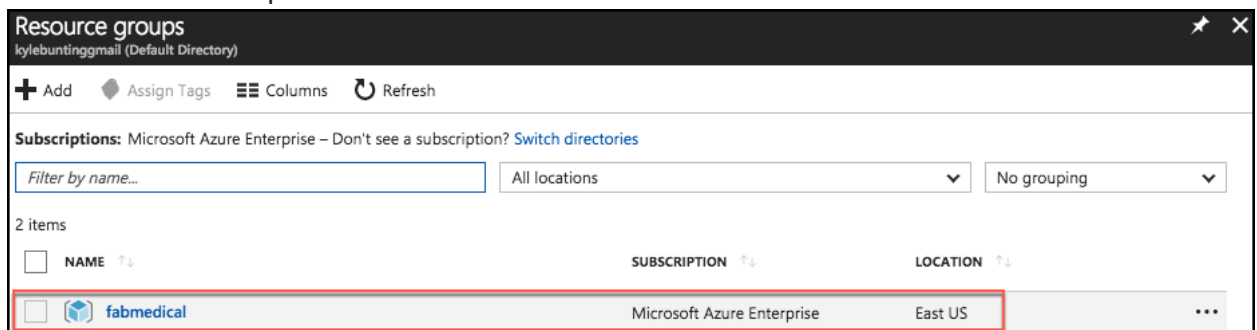


5. On the new Resource group blade, set the following:
 - a. Resource group name: Enter something like "fabmedical-SUFFIX," as shown in the following screen shot.
 - b. Subscription: Select the subscription you will use for all the steps during the hackathon.
 - c. Resource group location: Choose a region where all Azure Container Registry SKUs are available, which is currently East US, West Central US, or West Europe, and remember this for future steps so that the resources you create in Azure are all kept within the same region.

d. Select **Create**.

Exit criteria

- Your Resource Group is listed in the Azure Portal.

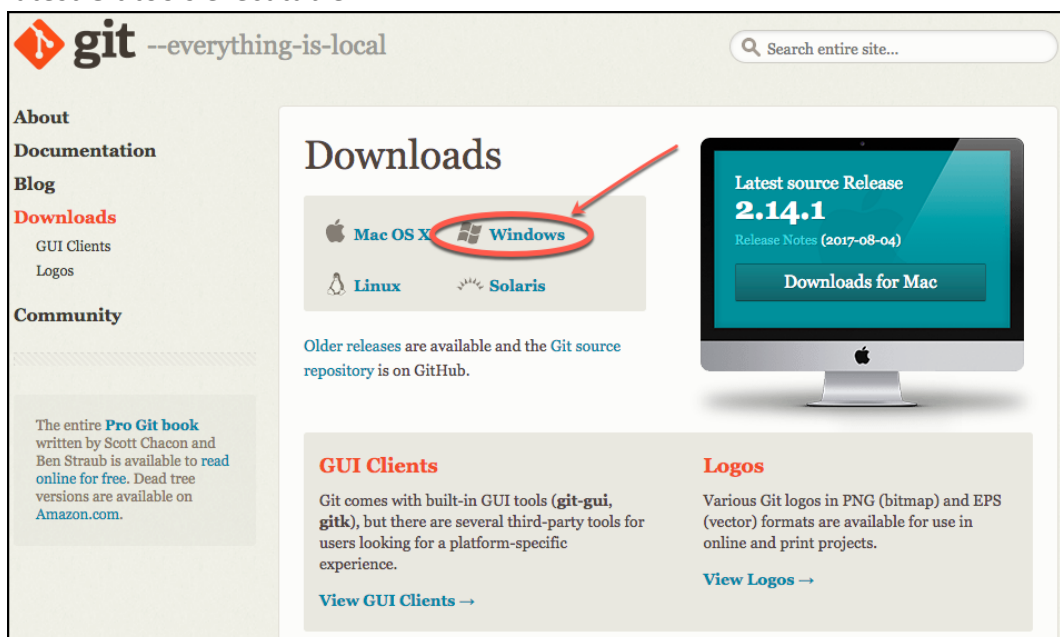


Task 2: Install Git tools

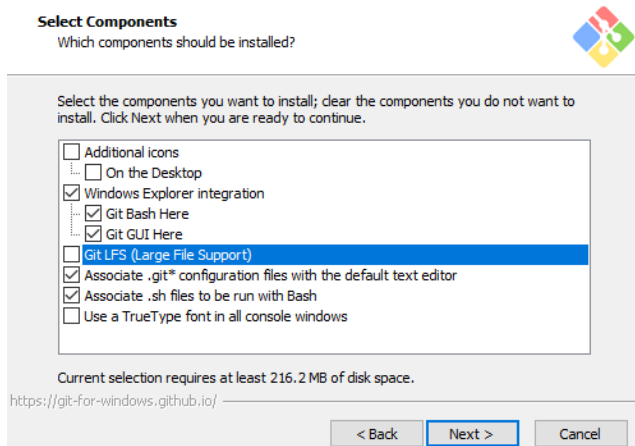
In this section, you will install Git tools. This documentation assumes you are installing Git tools for Windows but you may use another platform to achieve the same results at your own discretion.

Tasks to complete

1. Navigate to <https://git-scm.com/download>, and select Windows. This will download the latest Git tools executable.





2. Run the executable, selecting Next to pass the license dialog, and accept the default Git folder path.
3. On each dialog, choose the settings as shown in the following screen shots, and select Next until reaching the final screen where you will select Install.



Select Start Menu Folder


Where should Setup place the program's shortcuts?





Setup will create the program's shortcuts in the following Start Menu folder.

To continue, click Next. If you would like to select a different folder, click Browse.



Browse...

☐ Don't create a Start Menu folder

<https://git-for-windows.github.io/>

< Back

Next >

Cancel

Adjusting your PATH environment

How would you like to use Git from the command line?



☐ Use Git from Git Bash only

This is the safest choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

☒ Use Git from the Windows Command Prompt

This option is considered safe as it only adds some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from both Git Bash and the Windows Command Prompt.

☐ Use Git and optional Unix tools from the Windows Command Prompt

Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.

<https://git-for-windows.github.io/>

< Back

Next >

Cancel

Choosing HTTPS transport backend

Which SSL/TLS library would you like Git to use for HTTPS connections?

☒ Use the OpenSSL library

Server certificates will be validated using the ca-bundle.crt file.

☐ Use the native Windows Secure Channel library

Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.

https://git-for-windows.github.io/

< Back

Next >

Cancel

Configuring the line ending conversions

How should Git treat line endings in text files?

☒ Checkout Windows-style, commit Unix-style line endings

Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").

☐ Checkout as-is, commit Unix-style line endings

Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").

☐ Checkout as-is, commit as-is

Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").

https://git-for-windows.github.io/

< Back

Next >

Cancel

Configuring the terminal emulator to use with Git Bash

Which terminal emulator do you want to use with your Git Bash?

☒ Use MinTTY (the default terminal of MSYS2)

Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via 'winpty' to work in MinTTY.

☐ Use Windows' default console window

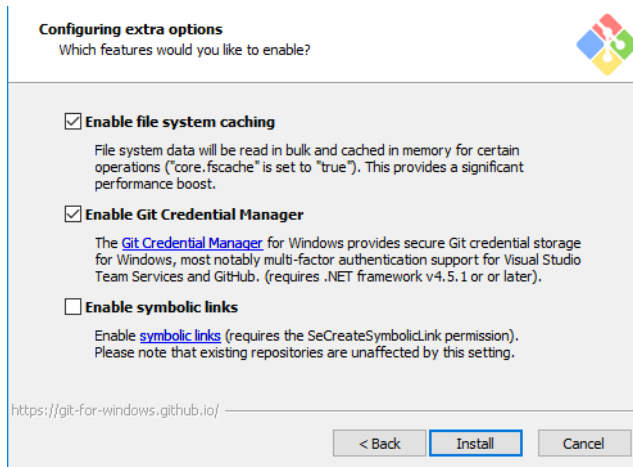
Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.

https://git-for-windows.github.io/

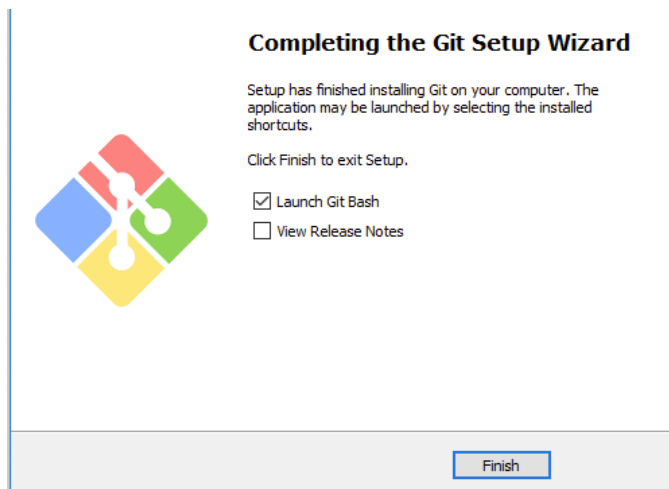
< Back

Next >

Cancel

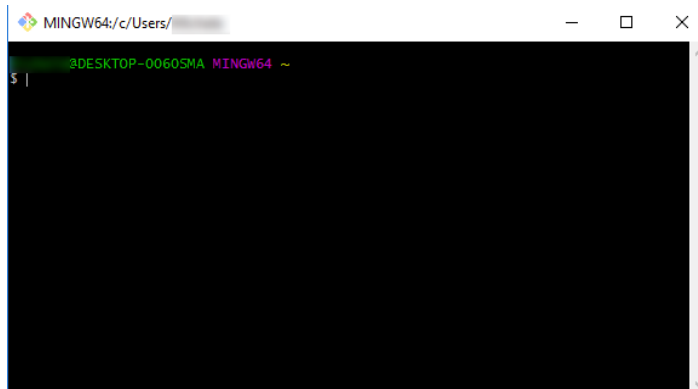


- When the installation is completed, check Launch Git Bash, and select Finish to show the command line window.



Exit criteria

- At the end of these steps you should see the Git Bash command line window.

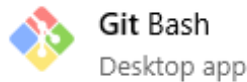


Task 3: Create an SSH key

In this section, you will create an SSH key to securely access the VMs you create during the upcoming exercises.

Tasks to complete

1. Open Git Bash to access the command line tool.



2. From the command line enter the following command to generate an SSH key pair. You can replace "admin" with your preferred name or handle.

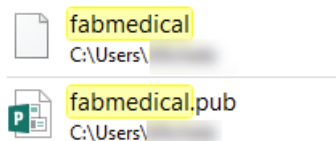
```
ssh-keygen -t RSA -b 2048 -C admin@fabmedical
```

3. You will be asked to save the generated key to a file. Enter "fabmedical" for the name.
4. Enter a passphrase when prompted, and don't forget it!
5. The file will be generated in your user folder, where Git Bash opens by default, so be sure to take note of the path.

A screenshot of a Git Bash terminal window. The title bar reads "MINGW64: c:/Users/[username]". The terminal shows the command `$ ssh-keygen -t RSA -b 2048 -C [username]@fabmedical` being executed. The output includes: "Generating public/private RSA key pair.", "Enter file in which to save the key (/c:/Users/[username]/.ssh/id_rsa): fabmedical", "Enter passphrase (empty for no passphrase):", "Enter same passphrase again:", "Your identification has been saved in fabmedical.", "Your public key has been saved in fabmedical.pub.", "The key fingerprint is:", "SHA256:EGip50pXkRTLVSis/Y8ILo/zeqjL3Gh6Vak94NJ+vIU [username]@fabmedical", "The key's randomart image is:", followed by a large ASCII art representation of the key. The terminal prompt at the bottom is `@DESKTOP-0060SMA MINGW64 ~`.

Exit criteria

- Navigate to your user folder at `c:\Users\[your username]`. You will see the file matching the name you provided.



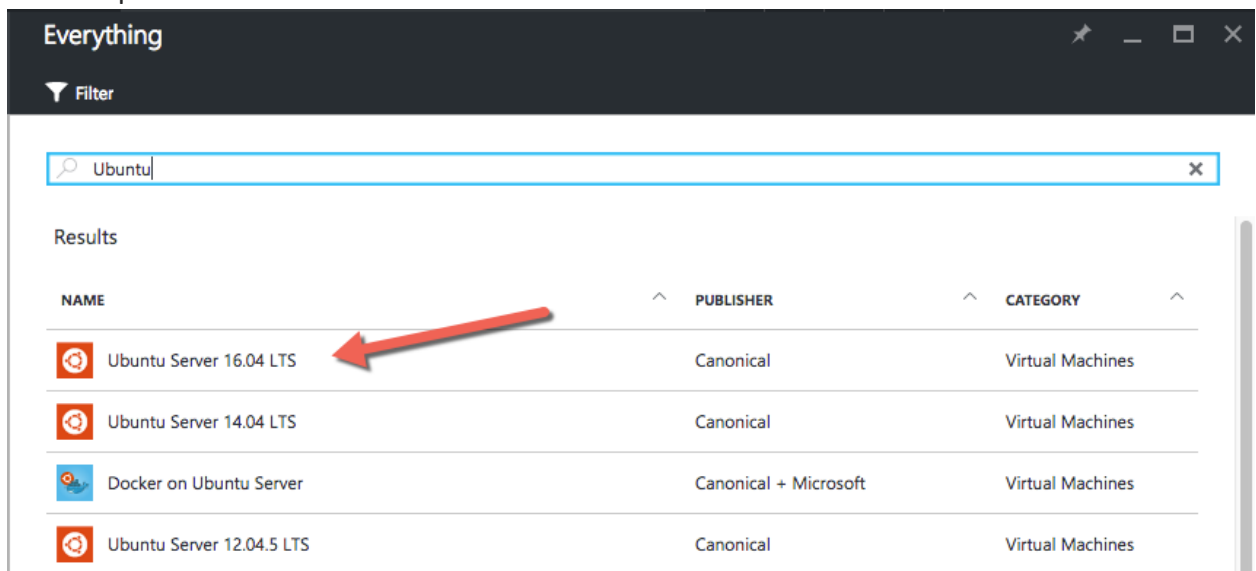
Task 4: Create a build agent VM

In this section, you will create a Linux VM to act as your build agent. You will be installing Docker to this VM once it is set up, and you will use this VM during the hackathon to develop and deploy.

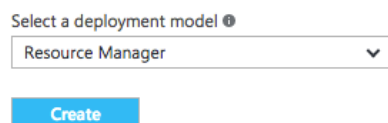
NOTE: You can set up your local machine with Docker however the setup varies for different versions of Windows. For this hackathon, the build agent approach simply allows for predictable setup.

Tasks to complete

1. From the Azure Portal, select **+ New**, type “Ubuntu” in the Search the marketplace text box and press Enter.



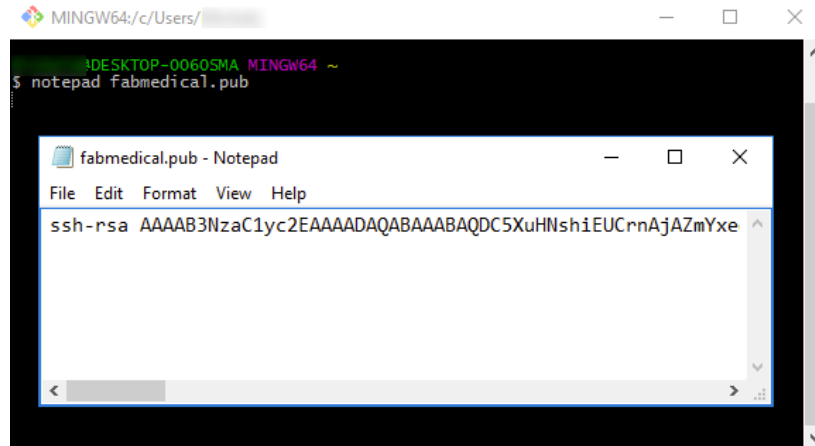
2. Select Ubuntu Server 16.04 LTS.
3. From the Ubuntu Server 16.04 LTS blade, select Resource Manager deployment model and select Create.



4. On the Basics blade of the Virtual Machine setup, set the following:
 - a. Name: Provide a unique name, such as “fabmedical-SUFFIX,” as shown in the following screenshot.
 - b. VM disk type: Leave as SDD.
 - c. User name: Provide a user name, such as “adminfabmedical”.
 - d. Authentication type: Leave as SSH public key.

- e. SSH public key: From your local machine, copy the public key portion of the SSH key pair you created previously, to the clipboard.
 - i. From Git Bash, verify you are in your user directory `c:\Users\[your username]`.
 - ii. Type the following command at the Git Bash prompt to open the public key that you generated in notepad.

```
notepad fabmedical.pub
```



- iii. Copy the entire contents of the file to the clipboard. Take care not to introduce any line breaks or spaces at the beginning or end of the text.
 - iv. Paste this value in the SSH public key textbox of the blade.
- f. Subscription: Choose the same subscription you are using for all your work.
- g. Resource group: Choose Use existing, and select the resource group you created previously.
- h. Location: Choose the same region that you did before.

- i. Select OK to complete the Basics blade.

Basics

*

Name

fabmedical-soll

✓

VM disk type ⓘ

SSD

▼

*

User name

adminfabmedical

✓

*

Authentication type

SSH public key Password

*

SSH public key ⓘ

Wrs0dGiUZWtdYTaq2tTPXpXCrY8MthjMe/
e78ZB8t5GGJr3C9jCoqBMsjPYjKPV3jg4ZWU
BpyK mzdd1oo4UUsMjB436qVpW7

✓

Subscription

Microsoft Azure Enterprise

▼

*

Resource group

☐ Create new
 ☒ Use existing

fabmedical

▼

*

Location

East US

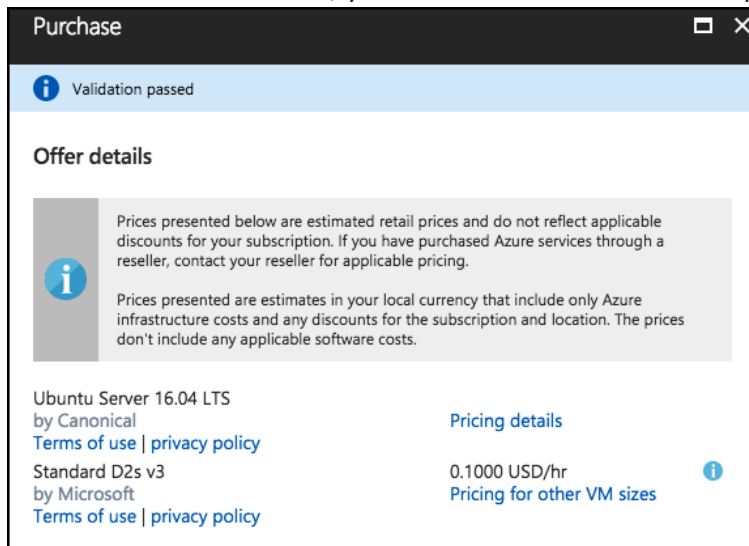
▼

5. From the Size blade choose D2S_V3 Standard, and click Select.

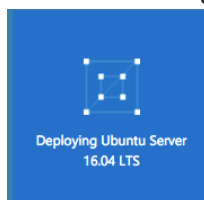
★ Recommended | [View all](#)

D2S_V3 Standard ★	D4S_V3 Standard ★	E2S_V3 Standard ★
2 vCPUs	4 vCPUs	2 vCPUs
8 GB	16 GB	16 GB
4 Data disks	8 Data disks	4 Data disks
4000 Max IOPS	8000 Max IOPS	4000 Max IOPS
16 GB Local SSD	32 GB Local SSD	32 GB Local SSD
Premium disk support	Premium disk support	Premium disk support
Load balancing	Load balancing	Load balancing
74.40 USD/MONTH (ESTIMATED)	148.80 USD/MONTH (ESTIMATED)	98.95 USD/MONTH (ESTIMATED)

- From the Settings blade, accept the default values for all settings and select OK.
- From the Purchase blade, you should see that validation passed and select Purchase.



- The VM will begin deployment to your Azure subscription.



Exit criteria

- When the VM is provisioned you will see it in your list of resources belonging to the resource group you created previously.

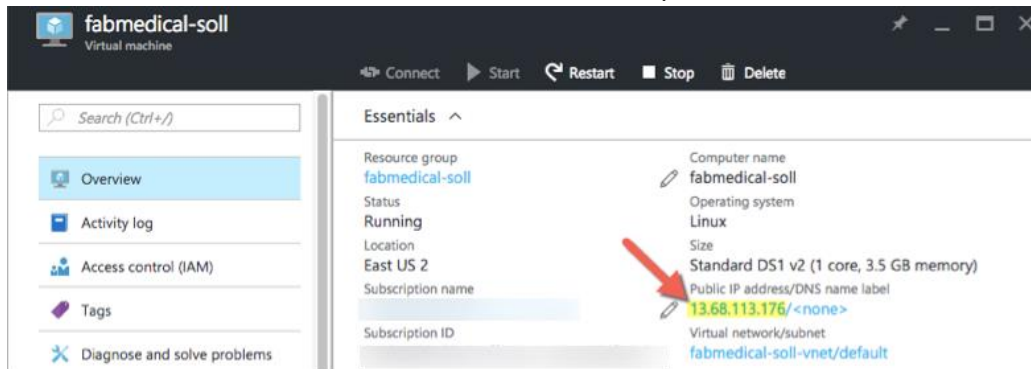
<input type="checkbox"/>	NAME ↑↓	TYPE ↑↓	LOCATION ↑↓	
<input type="checkbox"/>	fabmedicaldiag578	Storage account	East US	...
<input type="checkbox"/>	fabmedicalkyle	Container registry	East US	...
<input type="checkbox"/>	fabmedical-kyle	Virtual machine	East US	...
<input type="checkbox"/>	fabmedical-kyle_OsDisk_1_8631e96d054b42...	Disk	East US	...
<input type="checkbox"/>	fabmedical-kyle552	Network interface	East US	...
<input type="checkbox"/>	fabmedical-kyle-ip	Public IP address	East US	...
<input type="checkbox"/>	fabmedical-kyle-nsg	Network security group	East US	...

Task 5: Connect securely to the build agent

In this section, you will validate that you can connect to the new build agent VM.

Tasks to complete

1. From the Azure portal, navigate to the Resource Group you created previously and select the new VM, fabmedical-SUFFIX.
2. In the Essentials area for the VM, take note of the public IP address for the VM.



3. From your local machine, launch Git Bash and navigate to your user directory `c:\Users\[your username]` where the key pair was previously created.

`cd c:\users\<your username>\.ssh`
4. Connect to the new VM you created by typing the following command.

```
ssh -i [PRIVATEKEYNAME] [BUILDAGENTUSERNAME]@[BUILDAGENTIP]
```

Replace the bracketed values in the command as follows:

- [PRIVATEKEYNAME]: Use the private key name “fabmedical,” which was created above.
- [BUILDAGENTUSERNAME]: Use the username for the VM, such as adminfabmedical.
- [BUILDAGENTIP]: The IP address for the build agent VM, retrieved from the VM Overview blade in the Azure Portal.

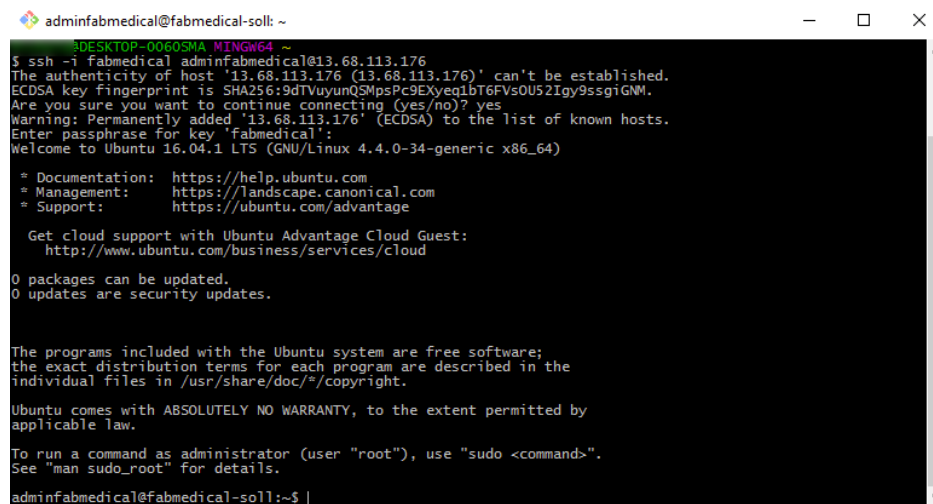
```
ssh -i fabmedical adminfabmedical@13.68.113.176
```

5. You will be asked to confirm if you want to connect, as the authenticity of the connection cannot be validated. Type “yes”.
6. You will be asked for the passphrase for the private key you created previously. Enter this value.

Exit criteria

- You will now be connected to the VM with a command prompt such as the following. Keep this command prompt open for the next step.

adminfabmedical@fabmedical-soll:~\$



```
adminfabmedical@fabmedical-soll: ~
$ ssh -i Fabmedical adminfabmedical@13.68.113.176
The authenticity of host '13.68.113.176 (13.68.113.176)' can't be established.
ECDSA key fingerprint is SHA256:9dTUyUunQSMpsPc9EXyeq1bT6FV50U52Igy9ssgiGM.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '13.68.113.176' (ECDSA) to the list of known hosts.
Enter passphrase for key 'Fabmedical':
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-34-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud

0 packages can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

adminfabmedical@fabmedical-soll:~$
```

NOTE: If you have issues connecting, you may have pasted the SSH public key incorrectly. Unfortunately, if this is the case, you will have to recreate the VM and try again.

Task 6: Complete the build agent setup

In this task you will update the packages and install Docker engine.

Tasks to complete

1. Continue in Git Bash with the SSH connection open to the build agent VM.
2. Update the Ubuntu packages and install Docker engine, curl, node.js and the node package manager in a single step by typing the following in a single line command.

```
sudo apt-get update && sudo apt install docker.io curl nodejs npm
```

3. Now, upgrade the Ubuntu packages to the latest version by typing the following in a single line command.

```
sudo apt-get upgrade
```

4. When the command has completed, check the Docker version installed by executing this command. The output may look something like that shown in the following screen shot. Note that the server version is not shown yet, because you didn't run the command with elevated privileges (to be addressed shortly).

```
docker version
```

```
adminfabmedical@fabmedical-soll: ~  
adminfabmedical@fabmedical-soll:~$ docker version  
Client:  
Version:      1.12.1  
API version:  1.24  
Go version:   go1.6.2  
Git commit:   23cf638  
Built:        Tue, 27 Sep 2016 12:25:38 +1300  
OS/Arch:      linux/amd64  
Cannot connect to the Docker daemon. Is the docker daemon running on this host?  
adminfabmedical@fabmedical-soll:~$ |
```

5. You may check the versions of node.js and npm as well, just for information purposes, using these commands.

```
nodejs --version  
npm -version
```

6. Add your user to the Docker group so that you do not have to elevate privileges with sudo for every command. You can ignore any errors you see in the output.

```
sudo usermod -aG docker $USER
```

NOTE: You should type in the above command, as copy and paste can result in the “Usage: usermod” instructions being displayed. You may see the following error messages when running the sudo command. They can be ignored.

```
adminfabmedical@fabmedical-soll: ~  
adminfabmedical@fabmedical-soll:~$ sudo usermod -aG docker $USER  
sent invalidate(passwd) request, exiting  
sent invalidate(group) request, exiting  
sent invalidate(passwd) request, exiting  
sent invalidate(group) request, exiting
```

7. In order for the user permission changes to take effect, exit the SSH session by typing ‘exit’, then press <Enter>. Repeat the commands in Task 5 from step 4 to establish the SSH session again.
8. Run the Docker version command again, and note the output now shows the server version as well.

```
adminfabmedical@fabmedical-soll: ~  
adminfabmedical@fabmedical-soll:~$ docker version  
Client:  
Version:      1.12.1  
API version:  1.24  
Go version:   go1.6.2  
Git commit:   23cf638  
Built:        Tue, 27 Sep 2016 12:25:38 +1300  
OS/Arch:      linux/amd64  
  
Server:  
Version:      1.12.1  
API version:  1.24  
Go version:   go1.6.2  
Git commit:   23cf638  
Built:        Tue, 27 Sep 2016 12:25:38 +1300  
OS/Arch:      linux/amd64  
adminfabmedical@fabmedical-soll:~$
```

Exit criteria

- Run a few Docker commands.

- One to see if there are any containers presently running

```
docker ps
```

- One to see if any containers exist whether running or not

```
docker ps -a
```

- In both cases, you will have an empty list but no errors running the command. Your build agent is ready with Docker engine running properly.

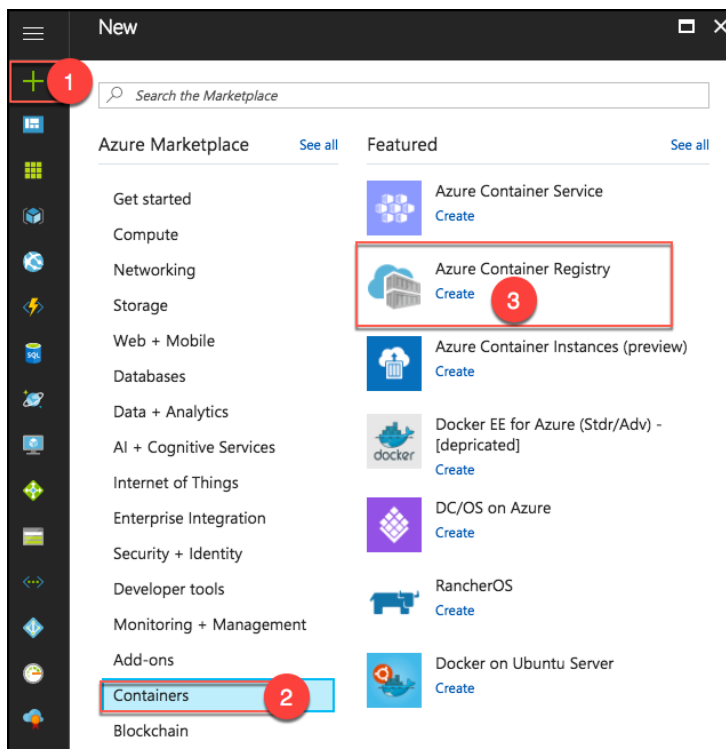
```
adminfabmedical@fabmedical-soll: ~
adminfabmedical@fabmedical-soll:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
adminfabmedical@fabmedical-soll:~$
```

Task 7: Create an Azure Container Registry

Docker images are deployed from a registry. To complete the hackathon, you will need access to a registry that is accessible to the Azure Container Service cluster you are creating. In this task, you will create an Azure Container Registry (ACR) for this purpose, where you push images for deployment.

Tasks to complete

1. In the [Azure Portal](#), select + New, Containers, Azure Container Registry, and select Create.



2. On the Create container registry blade, enter the following:
 - a. Registry name: Enter a name, such as "fabmedicalSUFFIX," as shown in the following screenshot.
 - b. Subscription: Choose the same subscription you are using for all your work.

- c. Resource group: Choose Use existing, and select the resource group you created previously.
- d. Location: Choose the same region that you did before.
- e. Admin user: Select Enable.
- f. SKU: Select Standard (Preview).

- g. Select Create

Exit criteria

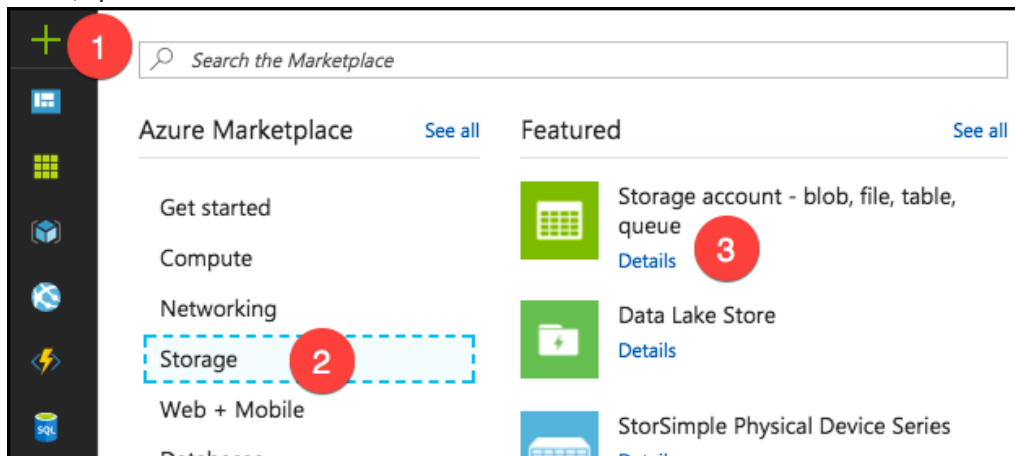
- Navigate to your ACR account in the Azure Portal. As this is a new account, you will not see any repositories yet. You will create these during the hackathon.

Task 8: Create a Storage Account

In this section, you will create an Azure Storage Account for storing your ACR account credentials, for use later by DC/OS.

Tasks to complete

1. In the Azure Portal, select + New, Storage, and then select Storage account – blob, file, table, queue.



2. On the Create storage account blade, enter the values below, as in the sample screenshot.
 - a. Name: Enter a unique name, such as fabmedicalstoreSUFFIX
 - b. Deployment model: Leave set to Resource Manager
 - c. Account Kind: Leave set to General Purpose
 - d. Performance: Leave set to Standard
 - e. Replication: Select Locally-redundant storage (LRS)
 - f. Secure transfer required: Leave set to Disabled
 - g. Subscription: Select your subscription
 - h. Resource group: Choose Use existing, and select the same resource group you created above.

- i. Location: Select the same location you used in previous steps

Create storage account

* Name fabmedicalstore ✓
.core.windows.net

Deployment model **Resource manager** Classic

Account kind General purpose

Performance **Standard** Premium

Replication Locally-redundant storage (LRS)

* Secure transfer required Disabled Enabled

* Subscription Microsoft Azure Enterprise

* Resource group
☐ Create new ☒ Use existing
fabmedical

* Location East US

Virtual networks (Preview)
Configure virtual networks Disabled Enabled

☐ Pin to dashboard

Create Automation options

- j. Select Create

3. Once provisioned, navigate to the newly created storage account in the Azure Portal.
4. Click on Blobs under Services on the Overview blade.

Open in Explorer → Move Delete storage account

Essentials ^

Resource group (change) fabmedical

Status Primary: Available

Location East US

Subscription (change) Microsoft Azure Enterprise

Subscription ID e8585ab7-56de-483f-a078-7e01bbb5cece

Performance Standard

Replication Locally-redundant storage (LRS)

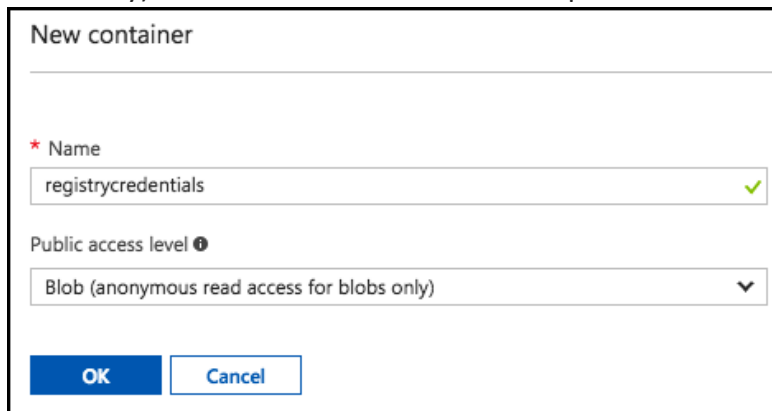
Services

Blobs
Object storage for understanding data
View metrics
Configure CORS rules
Setup custom domain

Files
File shares that use SMB 3.0 protocol
View metrics
Configure CORS rules

5. Select + Container on the Blob Service blade to create a new container.

6. In the New container dialog, enter a name and select Blob (anonymous read access for blobs only) from the Public access level drop down.



7. Select OK.

Exit criteria

- You have a storage account with a blob container defined.

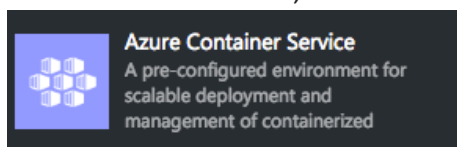
Task 9: Create an Azure Container Service cluster

In this task, you will create your Azure Container Service cluster based on Apache Mesos. You will use the same SSH key you created previously to connect to this cluster in the next task.

NOTE: In this step, you will be asked to create a second Resource Group as part of the template for creating the Azure Container Service cluster. This is due to the current template not providing the option to select an *existing* Resource Group, and this may change in the future.

Tasks to complete

1. From the Azure Portal, select **+ New**, Containers and select **Azure Container Service**.



2. From the Azure Container Service blade select Create.
3. In the Basics blade provide the information shown in the screen shot that follows:
 - a) Name: Enter fabmedical-SUFFIX
 - b) Subscription: choose your subscription
 - c) Resource group: Create new and provide a unique name. Since the template does not support using an existing Resource Group, provide a new name such adding a "2" to the suffix, such as fabmedical-SUFFIX2.
 - d) Location: Choose a location

Basics

* Name
fabmedical-acis ✓

Subscription
Microsoft Azure Enterprise ▼

Resource group
☒ Create new ☐ Use existing
fabmedical2 ✓

* Location
East US ▼

2. Select OK.
3. On the Master Configuration blade provide:
 - a) Orchestrator: DC/OS
 - b) DNS Name Prefix: Set the DNS prefix to something like “fabmedical-SUFFIX”.
User name: Enter a username such as “adminfabmedical”.
 - c) SSH Public Key: Paste the same SSH public key you used for the agent VM previously.
 - d) Master Count: leave this at 1.
 - e) VM Diagnostics: leave disabled.

Master configuration

* Orchestrator ⓘ
 DC/OS

* DNS name prefix ⓘ
 fabmedical ✓

Master credentials

* User name ⓘ
 adminfabmedical ✓

* SSH public key ⓘ
 ssh-rsa
 AAAAB3NzaC1yc2EAAAADAQABAAQDJ
 dRKb/G2rbFtmbfVOq4Apz0FbfaEYJOZUay3r ✓

Settings

Master count ⓘ
 1

VM diagnostics ⓘ
 Disabled Enabled

4. Select OK.
5. On the Agent configuration blade, set the following settings:
 - a. Agent count: 2
 - b. Agent virtual size: Standard DS2

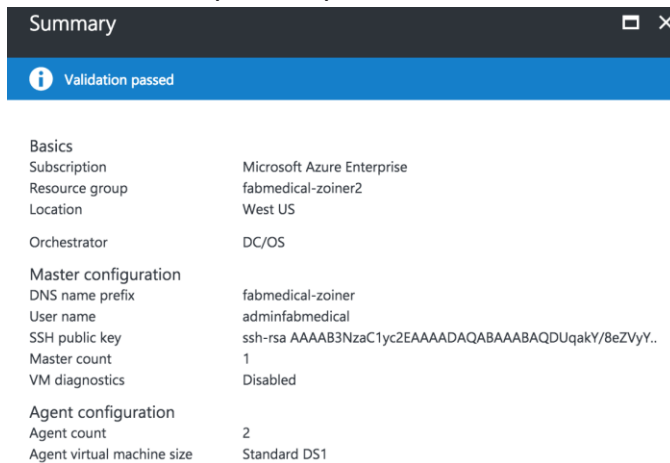
Agent configuration

* Agent count ⓘ
 2 ✓

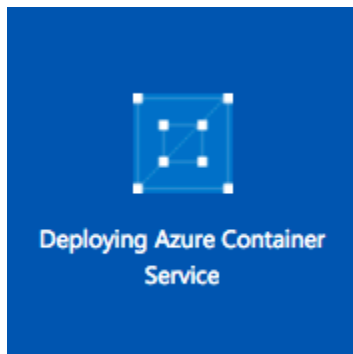
* Agent virtual machine size ⓘ
 2x Standard DS2 >

6. Select OK.

7. On the Summary blade, you should see that validation passed and select OK.

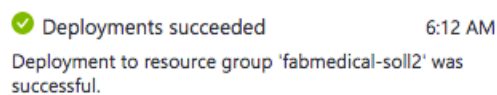


8. The Azure Container Service cluster will begin deployment to your Azure subscription.



Exit criteria

- You should see a successful deployment notification when the cluster is ready. It can take up to 10 minutes before your Azure Container Service cluster is listed in the Azure Portal.



Note: If you experience errors related to lack of available cores, you may have to delete some other compute resources or request additional cores be added to your subscription and then try this again.