

Container Workshop

Lab: Docker Local + ACS-Docker Swarm

Table of Contents

[Exercise 1: Working with Local Docker 3](#_Toc473533114)

[Task 1: Install Docker Toolbox 3](#_Toc473533115)

[Task 2: Download deployment files 3](#_Toc473533116)

[Task 3: Deploy the application to local Docker 3](#_Toc473533117)

[Exercise 2: Working with ACS – Docker Swarm 7](#_Toc473533118)

[Task 1: Create Secure Shell (SSH) keys 7](#_Toc473533119)

[Task 2: Create Docker Swarm cluster 7](#_Toc473533120)

[Task 3: Review Swarm master 8](#_Toc473533121)

[Task 4: Review Swarm agent cluster 9](#_Toc473533122)

[Task 5: Deploy the application to Swarm cluster 10](#_Toc473533123)

[Exercise 3: OMS Log Analytics for Docker 12](#_Toc473533124)

[Task 1: Setup OMS Log Analytics for Docker 12](#_Toc473533125)

[Exercise 4: Cleanup 16](#_Toc473533126)

[Task 1: Cleanup 16](#_Toc473533127)

# Exercise 1: Working with Local Docker

In this lab, you will use Docker Toolbox to deploy an application to your local Docker environment

Estimated time to complete this lab: **20 minutes**

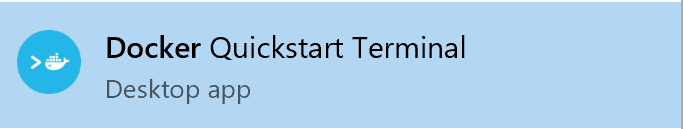
### Task 1: Install Docker Toolbox

1. If you do not have Docker Toolbox, please Install Docker Toolbox for Windows:

<https://docs.docker.com/toolbox/toolbox_install_windows/>

### Task 2: Download deployment files

1. Open Docker Quickstart Terminal



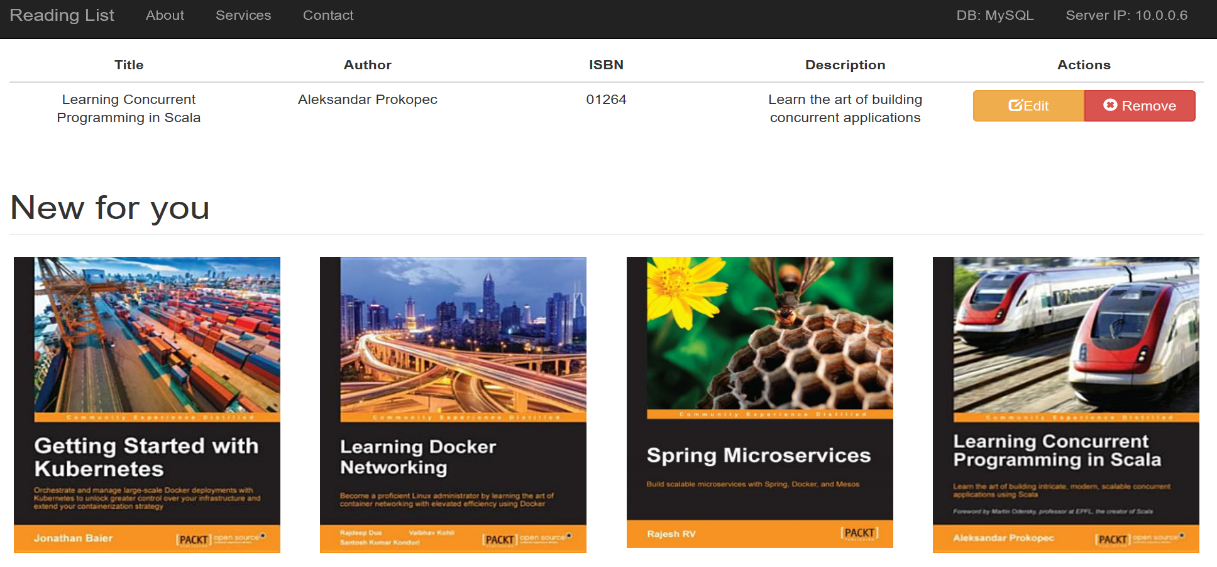
1. Clone the git repo containing the deployment files

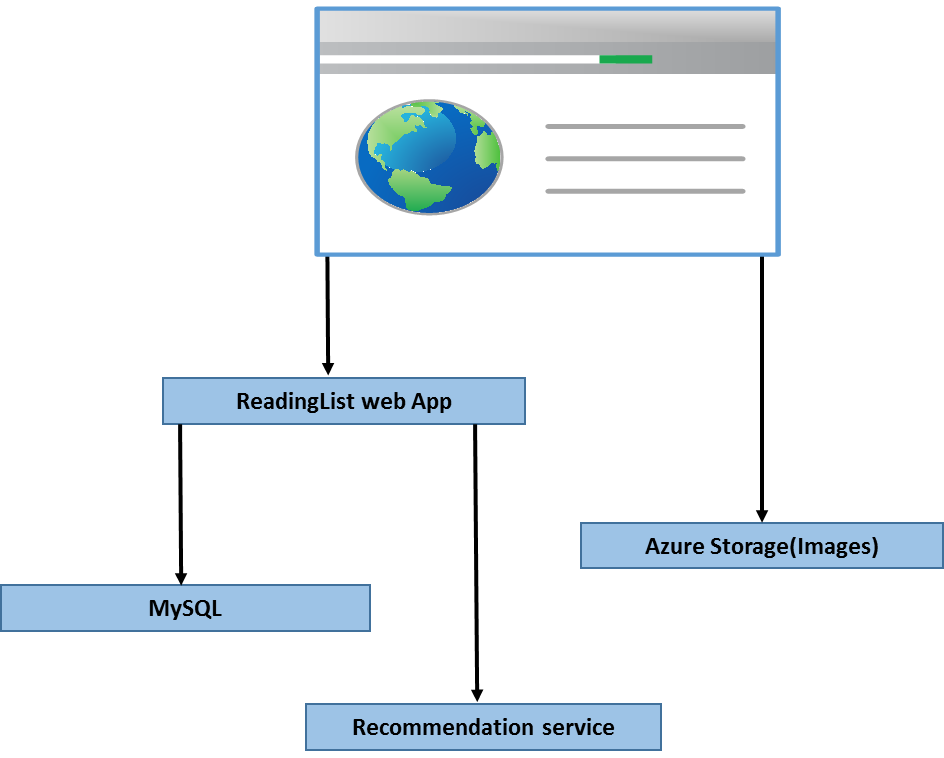
git clone <https://github.com/RaviTella/PCFAzureReadingListApplication>

### Task 3: Deploy the application to local Docker

The application you will be deploying is a multi-container application. It comprises of a web application called “ReadingList web app” which depends on “MySQL” for persisting reading lists and a RESTful web service called “Recommendation service” that recommends books. The recommendations include URLs for the book images. These images are served by Azure Storage.

At end of the task you will have 3 containers one each for ReadingList web application, MySQL and Recommendation service.





1. Make the docker directory your current directory

cd PCFAzureReadingListApplication/src/main/docker

1. There should be no images

Docker images

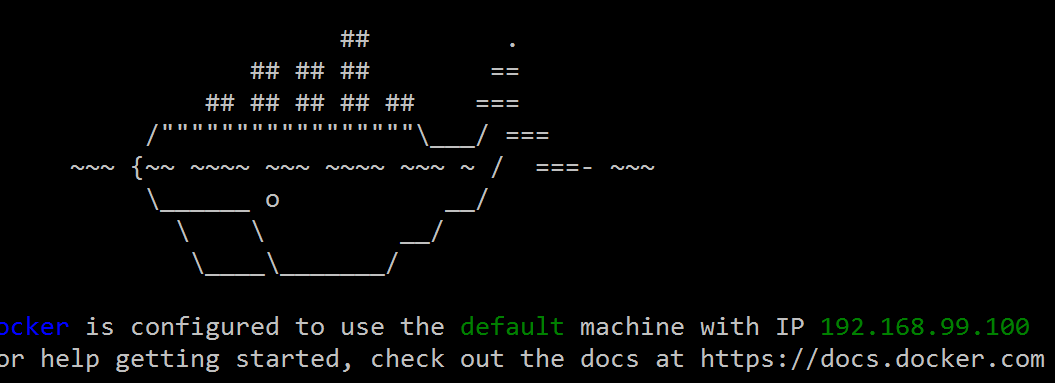
1. Deploy the application stack using docker-compose

docker-compose -p readinglist up -d

1. Access the application from a browser

http://192.168.99.100/readingList

IP address can be found at start of the Docker quickstart terminal



1. Check the number of services running for the deployed application stack

docker-compose -p readinglist ps

1. Check for images

docker images

1. Scale the service names “api”

docker-compose -p readinglist scale api=2

1. Check the number of services running for the deployed application stack

docker-compose -p readinglist ps

1. Scale the service names “web”, it will fail. Do you know why?

docker-compose -p readinglist scale web=2

1. Stop the application stack and remove the images

docker-compose -p readinglist down --rmi all

1. There should be no services running for the application stack

docker-compose -p readinglist ps

# Exercise 2: Working with ACS – Docker Swarm

In this lab, you will create a Docker Swarm cluster using Azure Container service and deploy the application stack to the Swarm cluster.

Estimated time to complete this lab: **60 minutes**

### Task 1: Create Secure Shell (SSH) keys

1. Open Git Bash



1. Generate ssh key pair (don’t set a passphrase, just hit enter key when asked for passphrase)

ssh-keygen -t rsa -b 2048 -C "acsadmin@acs" -f ~/.ssh/acs\_rsa

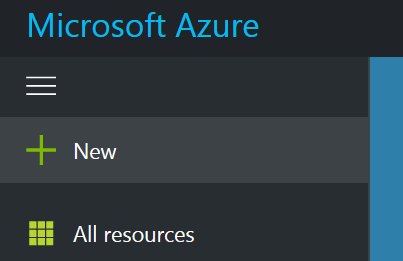
1. Verify the keys

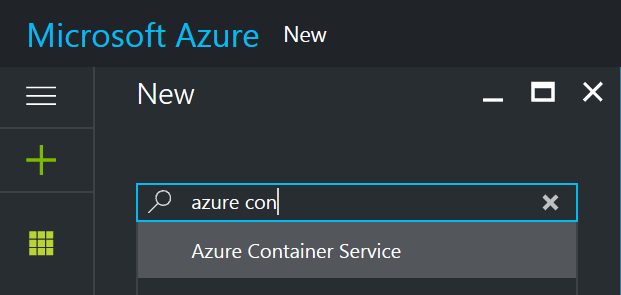
Location: C:\Users\<username>\.ssh

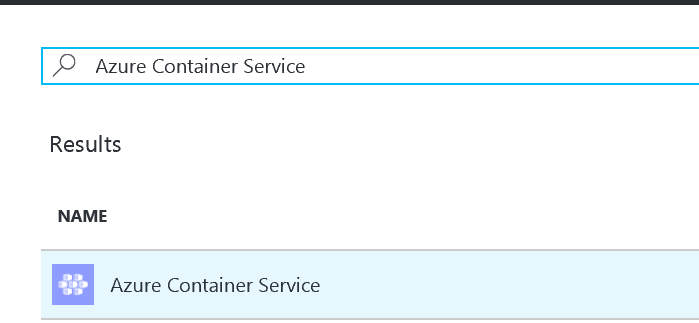
Keys: acs\_rsa.pub and acs\_rsa

### Task 2: Create Docker Swarm cluster

1. Create Docker Swarm
2. cluster via Azure portal



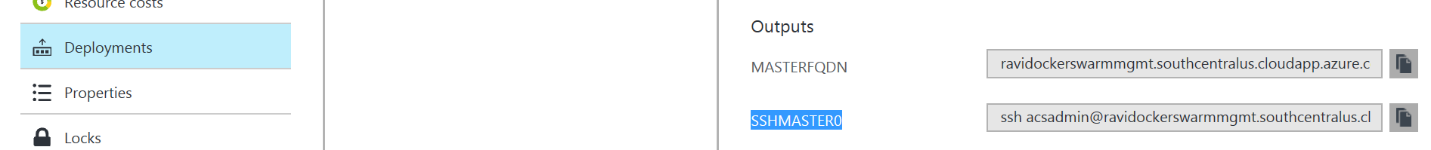




*Username:* acsadmin *SSH public Key:* content from acs\_rsa.pub (Content will start with ssh-rsa) *Resource group:* workshop-acsswarm\_rg *Location:* South Central US *Orchestrator:* Swarm *Agent count:* 3 *Master count:* 1 *DNS prefix:* <first2lettersfirstname><first5letterslastname>dockerswarm (has to be unique) *VM diagnostics:* enable

### Task 3: Review Swarm master

1. Connect to master
   1. Copy “sshMaster0” value from deployment section for your resource group and append “-i C:/Users/<username>/.ssh/acs\_rsa”



Example: ssh acsadmin@ravidockerswarmmgmt.southcentralus.cloudapp.azure.com -A -p 2200 -i C:/Users/<username>/.ssh/acs\_rsa

* 1. Save the command to a text file for future use
  2. Execute the command in Git Bash, you will be logged into master.

1. Verify images on master
   1. Execute the following command to review the containers running in the master

docker ps

* 1. Execute the following command to logout of master

exit

* 1. Close Git Bash

### Task 4: Review Swarm agent cluster

1. Open SSH tunnel to Swarm endpoint
   1. Add “-L 2375:localhost:2375” to the previous command used to connect to master:

Example: ssh -L 2375:localhost:2375 acsadmin@ravidockerswarmmgmt.southcentralus.cloudapp.azure.com -A -p 2200 -i C:/Users/<username>/.ssh/acs\_rsa

* 1. Execute the command in Git Bash to open a SSL tunnel. Keep Git Bash open for the rest of the lab.

1. Connect Docker QuickStart terminal to Swarm cluster
   1. Open Docker Quickstart Terminal
   2. Execute the following commands to point local Docker client to remote cluster

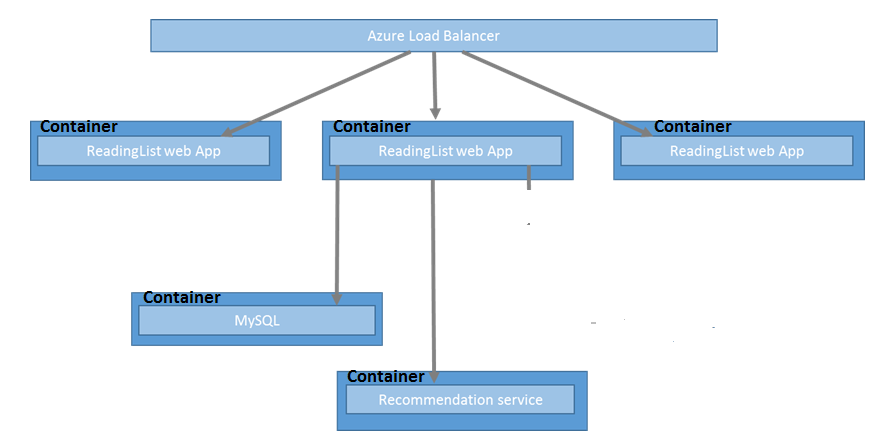
export DOCKER\_HOST=:2375 unset DOCKER\_TLS\_VERIFY unset DOCKER\_CERT\_PATH

1. Execute the following command to list images, there will be none:

docker images

### Task 5: Deploy the application to Swarm cluster

In this task, you will deploy the readinglist application stack to Swarm cluster and scale the reading list web app to 3 instances. You will end up with 5 containers. Three for reading list web app and one each for MySQL and recommendation service



* 1. Execute the following command to deploy the application stack

docker-compose -p readinglist up -d

* 1. Check the number of services running for the deployed application stack

docker-compose -p readinglist ps

* 1. Scale the service named “web”.

docker-compose -p readinglist scale web=3

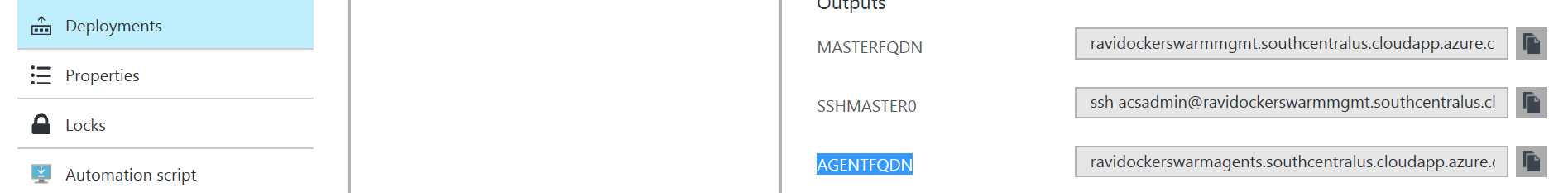
* 1. Check for the second instance of web

docker-compose -p readinglist ps

* 1. Access the application from a browser

http://< agentFQDN>/readinglist

“agentFQDN” can be found in the deployments section of your resource group. This is the DNS name used Azure

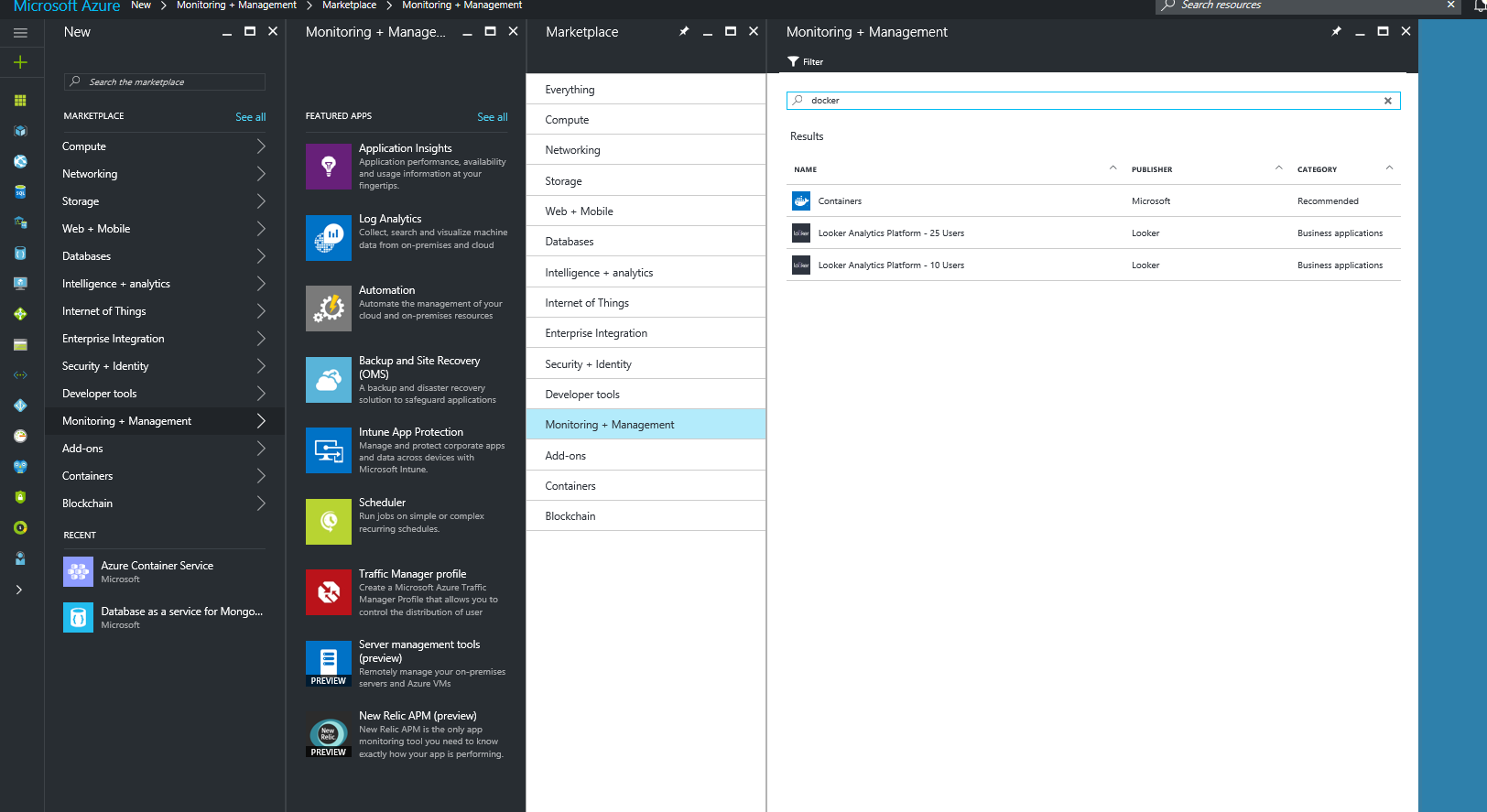


# Exercise 3: OMS Log Analytics for Docker

In this lab, you will set up OMS Log analytics for monitoring your Docker environment

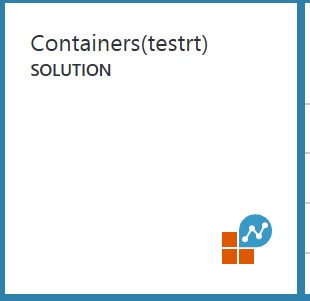
### Task 1: Setup OMS Log Analytics for Docker

1. Add Docker management solution
   1. sign in to the Azure portal.
   2. In the New blade under Marketplace, select Monitoring + management.
   3. In the Monitoring + management blade, click See all and search for Docker

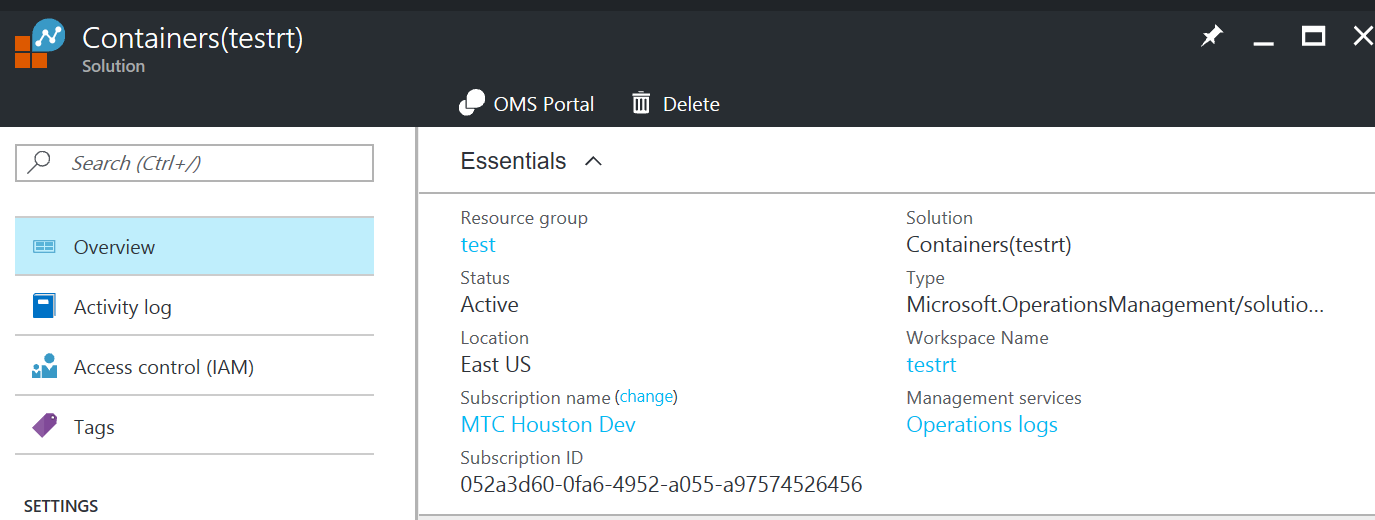


* 1. Click on Docker and then click create
  2. Click on select a workspace and then click create new workspace
  3. Provide the details for the new workspace, check “Pin to dashboard” and click OK
  4. After the validation click create

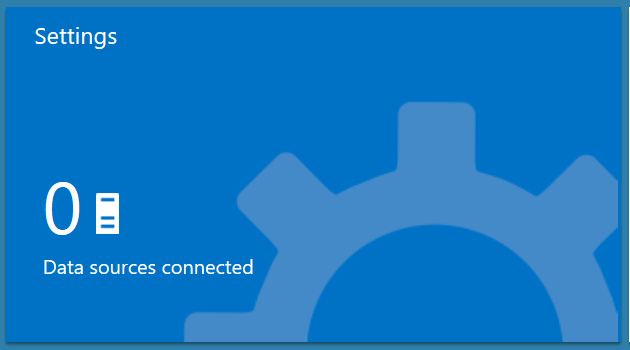
1. Retrieve the workspace id and primary key
   1. On the Azure portal dashboard click on the Containers solution tile

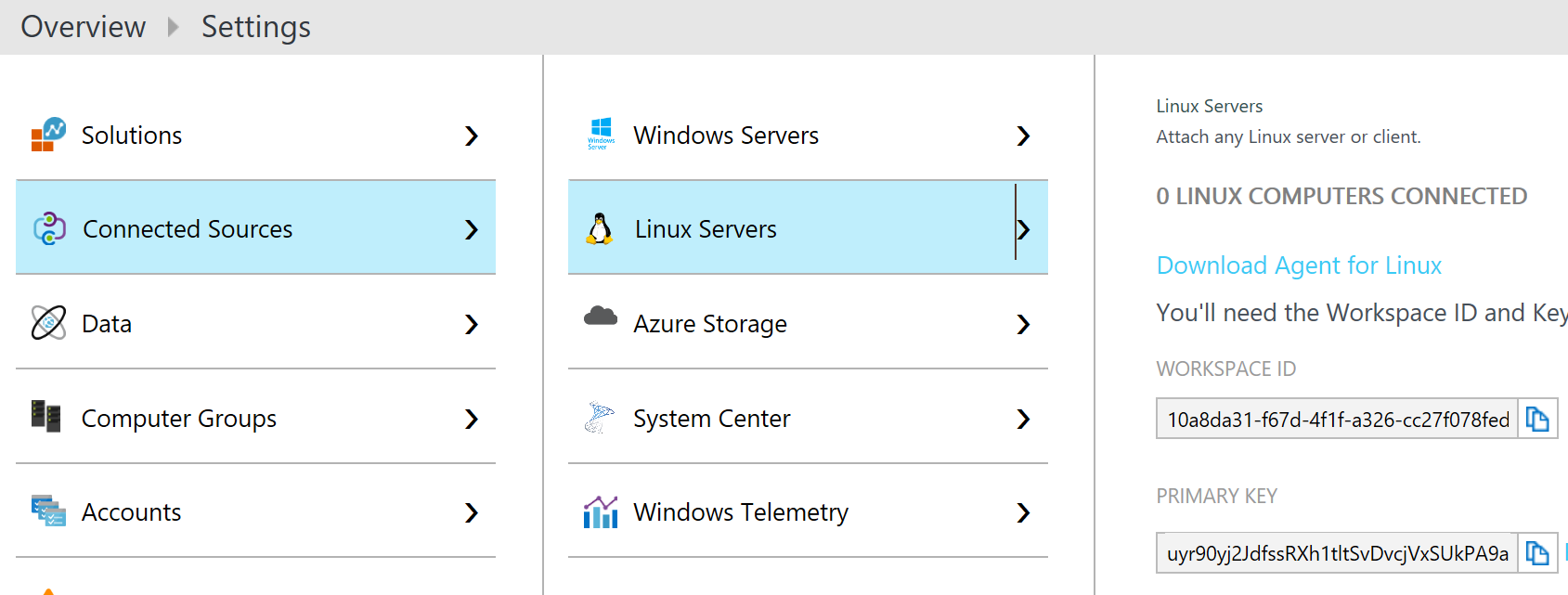


* 1. Lick on the OMS Portal icon at the top



* 1. Click on data sources connected tile on the home page and then open connected sources and then click on Linux servers





* 1. Copy workspace id and primary key

1. Install OMS agent containers on Docker nodes
2. Execute the following command in git bash terminal that was used to open the ssh tunnel
   1. Git clone <https://github.com/RaviTella/OMS-AppInsights>
3. Make oms the current directory

cd OMS-AppInsights/oms/

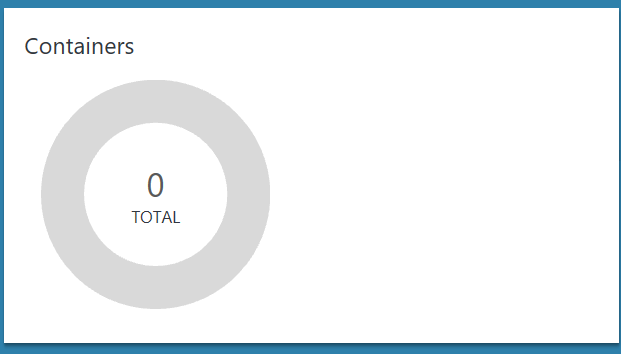
1. Update docker-compose.yml with workspace id and key
2. Execute he following command
   1. docker-compose -p omsagent up -d
3. Execute the following command to point git bash to swarm endpoint
   1. export DOCKER\_HOST=:2375
4. Execute the following command to deploy oms agent

docker-compose -p omsagent up -d

1. Execute the following command to scale oms agent to all the 3 agent nodes

docker-compose -p omsagent scale omsagent=3

1. Check OMS long analytics Docker view after a few minutes by clicking on the Containers tile on the OMS portal home page



# Exercise 4: Cleanup

Optionally, cleanup the resources you have created in this lab

### Task 1: Cleanup

1. Type the following command in Git Bash

exit

1. Close Git Bash
2. Delete the resource group for the cluster. This will delete all the resources you have created in this lab.

