Contents

[Prerequisites for this lab 3](#_Toc451870288)

[1. Activate your Azure Pass 4](#_Toc451870289)

[Notes about Monetary Credit 4](#_Toc451870290)

[2. Install and Get to know the Cross-Platform Command-Line Interface 5](#_Toc451870291)

[Use an installer for the Cross Platform CLI 5](#_Toc451870292)

[Connect to an Azure subscription from the Azure Command-Line Interface (Azure CLI) 5](#_Toc451870293)

[Use azure login to authenticate interactively 6](#_Toc451870294)

[Use azure login with a username and password 7](#_Toc451870295)

[Multiple subscriptions 7](#_Toc451870296)

[CLI command modes 8](#_Toc451870297)

[Storage of CLI settings 9](#_Toc451870298)

[Logging out 9](#_Toc451870299)

[3. Using Azure CLI commands in Resource Manager mode 9](#_Toc451870300)

[Setting the Resource Manager mode 9](#_Toc451870301)

[Imperative and declarative approaches 9](#_Toc451870302)

[azure account: Manage your account information 9](#_Toc451870303)

[azure config: commands to manage your local settings 9](#_Toc451870304)

[azure feature: commands to manage account features 9](#_Toc451870305)

[azure group: Commands to manage your resource groups 9](#_Toc451870306)

[azure location: Commands to get the available locations for all resource types 9](#_Toc451870307)

[azure network: Commands to manage network resources 9](#_Toc451870308)

[azure provider: Commands to manage resource provider registrations 9](#_Toc451870309)

[azure storage: Commands to manage your Storage objects 9](#_Toc451870310)

[azure tag: Commands to manage your resource manager tag 9](#_Toc451870311)

[azure vm: Commands to manage your Azure Virtual Machines 9](#_Toc451870312)

[4. Using Azure CLI Commands to create a Linux VM 9](#_Toc451870313)

[Method 1: Using the Azure CLI command Azure VM Create: 9](#_Toc451870314)

[Method 2: Using the Azure CLI command Azure VM Quick-Create: 9](#_Toc451870315)

[5. Using ARM templates to further automate VM Deployments 9](#_Toc451870316)

[Use the Azure CLI for Mac, Linux, and Windows with Azure Resource Manager 9](#_Toc451870317)

[Azure resources 10](#_Toc451870318)

[Authentication 10](#_Toc451870319)

[Set the Resource Manager mode 10](#_Toc451870320)

[Create a resource group 11](#_Toc451870321)

[Use resource group templates 11](#_Toc451870322)

[Creating a 5 node VMM scale set running an Apache/PHP website. 11](#_Toc451870323)

[Deploy this template to your Azure resource group 12](#_Toc451870324)

[Export a resource template 13](#_Toc451870325)

[6. Installing and using GIT as a repository 16](#_Toc451870326)

[Install NODE.JS 16](#_Toc451870327)

[Installing Node.js 16](#_Toc451870328)

[Testing Your Install 16](#_Toc451870329)

[Installing and Setting up Git 17](#_Toc451870330)

[Installing Git 17](#_Toc451870331)

[Signing up for a Free GitHub account 18](#_Toc451870332)

[Forking a Repository to your Account 18](#_Toc451870333)

[Cloning the Repository to Your Machine 18](#_Toc451870334)

[Creating a new Branch for your repo 19](#_Toc451870335)

[Staging your Changes for a Commit 20](#_Toc451870336)

[Committing your Changes 20](#_Toc451870337)

[Pushing your new Branch to Your Fork on GitHub 20](#_Toc451870338)

# Prerequisites for this lab

A Microsoft ID

An active Microsoft Azure Subscription

Microsoft Azure Cross platform CLI (for Windows, OSX or Linux)

Install the JQ JSON parser from <https://stedolan.github.io/jq/download/>

npm install azure-cli -g

**Windows example:**

Install chocolatey from a NuGet package <https://chocolatey.org/install>

Windows Instructions:

From an elevated powershell command line (right click the powershell icon and choose run as administrator) , run :

Set-executionpolicy unrestricted -force

Then install chocolatey by typing

iwr cin.st | iex

Visual studio Code

<https://code.visualstudio.com/>

PuTTY and PuTTYgen - [www.putty.org](http://www.putty.org)

Git: choco install git

Or

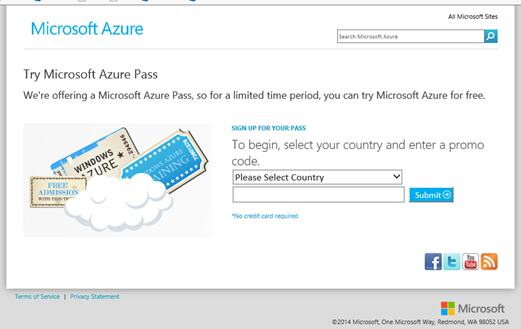
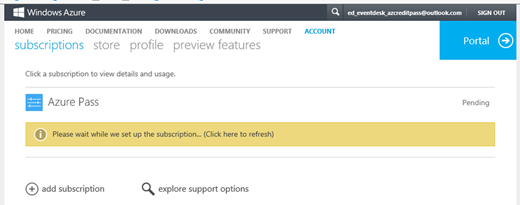
<https://central.github.com/mac/latest>

<https://github-windows.s3.amazonaws.com/GitHubSetup.exe>

<https://github.com/Microsoft/Git-Credential-Manager-for-Windows>

# Activate your Azure Pass

Redeeming Azure Monetary Credit for code <<**Azure Pass Code>>**

1. Go to <http://microsoftazurepass.com>. Select your country, type in the provided promo code and click “Submit.”  
   
2. Sign into a Microsoft account to continue. If you have an account password saved or you automatically sign into other Microsoft services that Microsoft account will be used to redeem the promo code.
3. The subscription will take 3-10 minutes to activate.  
   

## Notes about Monetary Credit

* Accounts are activated within minutes of the promo code being redeemed
* Users can view the cost of account usage
* User can enable public preview features
* Accounts cannot be extended - the duration and monetary credit amount cannot be changed
* Only one promo code can be redeemed per the life of a Microsoft account (LiveID)
  + Users who have used an Azure Trial can redeem a promo code. Users are able to redeem one trial and one promo code per account.
  + If a user has already redeemed a promo code, he/she will need to create a new Microsoft account.

For more detailed instructions on how to activate your Azure monetary credit, please visit <https://www.microsoftazurepass.com/howto>

# Install and Get to know the Cross-Platform Command-Line Interface

You will start by configuring your Azure subscription in the xplat-cli.

Quickly install the Azure Command-Line Interface (Azure CLI) to use a set of open-source shell-based commands for creating and managing resources in Microsoft Azure. You have several installation choices: use one of the provided installer packages for different operating systems, install from an npm package, or install the Azure CLI as a container in a Docker host. For more options and background, see the project repository on GitHub <https://github.com/azure/azure-xplat-cli> .

Once the Azure CLI has been installed, you will be able to connect it with your Azure subscription and run the **azure** commands from your command-line interface (Bash, Terminal, Command prompt, and so on) to work with your Azure resources.

## Use an installer for the Cross Platform CLI

The following installer packages are available on <https://azure.microsoft.com/en-us/downloads/> :

[Windows installer](https://www.microsoft.com/web/handlers/webpi.ashx?command=getinstallerredirect&appid=windowsazurexplatcli&mode=new) (<https://www.microsoft.com/web/handlers/webpi.ashx?command=getinstallerredirect&appid=windowsazurexplatcli&mode=new>)

<OS X installer> (http://aka.ms/mac-azure-cli)

[Linux installer](http://aka.ms/linux-azure-cli) (http://aka.ms/linux-azure-cli)

## Connect to an Azure subscription from the Azure Command-Line Interface (Azure CLI)

The Azure CLI is a set of open-source, cross-platform commands for working with the Azure platform. This article describes ways to provide your Azure account credentials to connect the Azure CLI to your Azure subscription. If you haven't already installed the CLI, see Install the Azure CLI above. If you don't have an Azure subscription, you can create a [free account](http://azure.microsoft.com/free/) in just a couple of minutes (credit card required) or use a provided Microsoft Azure Pass.

* **Log in to Azure using a work or school account or a Microsoft account identity** - Use the azure login command in CLI version 0.9.10 and above with either type of account identity to authenticate through Azure Active Directory. The CLI (version 0.9.9 and above) also supports interactive authentication through a web portal for accounts that have enabled multi-factor authentication. Also use the azure login command to authenticate a service principal for an Azure Active Directory application, which is useful for running automated services. After logging in with a supported account identity, you can use either Azure Resource Manager mode or Azure Service Management mode commands.

##### **Note:**

If you are using a version of the Azure CLI that is prior to version 0.9.10, you can use the azure login command only with a work or school account; Microsoft account identities do not work. However, if you want, you can [create a work or school ID from your Microsoft account ID](https://azure.microsoft.com/en-us/documentation/articles/virtual-machines-windows-create-aad-work-id/). https://azure.microsoft.com/en-us/documentation/articles/virtual-machines-windows-create-aad-work-id/

For background about different account identities and Azure subscriptions, see [How Azure subscriptions are associated with Azure Active Directory](https://azure.microsoft.com/en-us/documentation/articles/active-directory-how-subscriptions-associated-directory/). https://azure.microsoft.com/en-us/documentation/articles/active-directory-how-subscriptions-associated-directory/

## Use azure login to authenticate interactively

Use the azure login command -- without any arguments -- to authenticate interactively with either:

a work or school account identity (also called an organizational account) that requires multi-factor authentication, or

a Microsoft account identity when you want to access Resource Manager mode commands

##### **Note:**

In both cases, authentication and authorization are performed using Azure Active Directory. If you use a Microsoft account identity, the log in process accesses your Azure Active Directory default domain. (If you signed up for a free Azure account, you might not be aware that Azure Active Directory created a default domain for your account.)

Interactively logging in is easy: type azure login and follow the prompts as shown below:

**azure login**

info: Executing command login

info: To sign in, use a web browser to open the page http://aka.ms/devicelogin. Enter the code XXXXXXXXX to authenticate. If you're signing in as an Azure AD application, use the --username and --password parameters.

Copy the code offered to you, above, and open a browser to http://aka.ms/devicelogin. Enter the code, and then you are prompted to enter the username and password for the identity you want to use. When that process completes, the command shell completes the log in process. It might look something like:

info: Added subscription Visual Studio Ultimate with MSDN

info: Added subscription Azure Free Trial

info: Setting subscription "Visual Studio Ultimate with MSDN" as default

+

info: login command OK

## Use azure login with a username and password

Use the azure login command with a username parameter or with both a username and a password to authenticate when you want to use a work or school account that doesn't require multi-factor authentication. The following example passes the username of an organizational account:

**azure login -u robakk@nlazurepro.onmicrosoft.com**

info: Executing command login

Password: \*\*\*\*\*\*\*\*\*

/info: Added subscription Internal

+

info: login command OK

Enter your password when prompted.

If this is your first time logging in with these credentials, you are asked to verify that you wish to cache an authentication token. This prompt also occurs if you have previously used the azure logout command (described later in the article). To bypass this prompt for automation scenarios, run azure login with the -q parameter.

## Multiple subscriptions

If you have multiple Azure subscriptions, connecting to Azure will grant access to all subscriptions associated with your credentials. One subscription is selected as the default, and used by the Azure CLI when performing operations. You can view the subscriptions, as well as which one is the default, using the azure account list command. This command returns information similar to the following:

Copy to clipboardCopy

info: Executing command account list

data: Name Id Current

data: ---------------- ------------------------------------ -------

data: Azure-sub-1 #################################### true

data: Azure-sub-2 #################################### false

In the above list, the **Current** column indicates the current default subscription as Azure-sub-1. To change the default subscription, use the azure account set command, and specify the subscription that you wish to be the default. For example:

Copy to clipboardCopy

azure account set Azure-sub-2

This changes the default subscription to Azure-sub-2.

##### **Note:**

Changing the default subscription takes effect immediately, and is a global change; new Azure CLI commands, whether you run them from the same command-line instance or a different instance, use the new default subscription.

If you wish to use a non-default subscription with the Azure CLI, but don't want to change the current default, you can use the --subscription option for the command and provide the name of the subscription you wish to use for the operation.

Once you are connected to your Azure subscription, you can start using the Azure CLI commands to work with Azure resources.

## CLI command modes

The Azure CLI provides two command modes for working with Azure resources, with different command sets:

* **Resource Manager mode** - for working with Azure resources in the Resource Manager deployment model. To set this mode, run azure config mode arm.
* **Service Management mode** - for working with Azure resources in the classic deployment model. To set this mode, run azure config mode asm.

When first installed, the CLI is in Service Management mode.

##### **Note:**

The Resource Manager mode and Service Management mode are mutually exclusive. That is, resources created in one mode cannot be managed from the other mode.

## Storage of CLI settings

Whether you log in with the azure login command or import publish settings, your CLI profile and logs are stored in a .azure directory located in your user directory. Your user directory is protected by your operating system; however, it is recommended that you take additional steps to encrypt your user directory. You can do so in the following ways:

On Windows, modify the directory properties or use BitLocker.

On Mac, turn on FileVault for the directory.

On Ubuntu, use the Encrypted Home directory feature. Other Linux distributions offer similar features.

## Logging out

To log out, use the following command:

**azure logout -u <username>**

If the subscriptions associated with the account were only authenticated with Active Directory, logging out deletes the subscription information from the local profile. However, if a publish settings file had also been imported for the subscriptions, logging out only deletes Active Directory related information from the local profile.

**Note:** If you prefer using the login method, use the following command:

azure login -u username -p password

Now that you have configured your Azure subscription in the command line you will proceed to create the virtual machine.

# Using Azure CLI commands in Resource Manager mode

This article provides syntax and options for Azure command line interface (CLI) commands you'd commonly use to create and manage Azure resources in the Azure Resource Manager deployment model. You access these commands by running the CLI in Resource Manager (arm) mode. This is not a complete reference, and your CLI version may show slightly different commands or parameters.

To get started, first [install the Azure CLI](https://github.com/Azure/azure-content/blob/master/articles/xplat-cli-install.md) and [connect to your Azure subscription](https://github.com/Azure/azure-content/blob/master/articles/xplat-cli-connect.md) by using a work or school account or a Microsoft account identity.

For current command syntax and options at the command line in Resource Manager mode, type azure help or, to display help for a specific command, azure help [command]. You'll also find CLI examples in the documentation for creating and managing specific Azure services.

Optional parameters are shown in square brackets (for example, [parameter]). All other parameters are required.

In addition to command-specific optional parameters documented here, there are three optional parameters that can be used to display detailed output such as request options and status codes. The -v parameter provides verbose output, and the -vv parameter provides even more detailed verbose output. The --json option will output the result in raw json format.

## Setting the Resource Manager mode

Use the following command to enable Azure CLI Resource Manager commands.

azure config mode arm

The Azure Resource Manager mode and Azure Service Management mode are mutually exclusive. That is, resources created in one mode cannot be managed from the other mode.

## Imperative and declarative approaches

The Resource Manager mode of the Azure CLI gives you commands that create resources imperatively on the command line. For example, if you type azure group create <groupname> <location>you are asking Azure to create a resource group, and with azure group deployment create <resourcegroup> <deploymentname> you are instructing Azure to create a deployment of any number of items and place them in a group. Because each type of resource has imperative commands, you can chain them together to create fairly complex deployments.

However, using resource group templates that describe a resource group is a declarative approach that is far more powerful, allowing you to automate complex deployments of (almost) any number of resources for (almost) any purpose. When using templates, the only imperative command is to deploy one. For a general overview of templates, resources, and resource groups, see [Azure Resource Group Overview](https://github.com/Azure/azure-content/blob/master/articles/resource-group-overview.md). https://github.com/Azure/azure-content/blob/master/articles/resource-group-overview.md

## azure account: Manage your account information

Your Azure subscription information is used by the tool to connect to your account.

**List the imported subscriptions**

account list [options]

**Show details about a subscription**

account show [options] [subscriptionNameOrId]

**Set the current subscription**

account set [options] <subscriptionNameOrId>

**Commands to manage your account environment**

account env list [options]

account env show [options] [environment]

## azure config: commands to manage your local settings

**List Azure CLI configuration settings**

config list [options]

**Delete a config setting**

config delete [options] <name>

**Update a config setting**

config set <name> <value>

**Sets the Azure CLI working mode to either arm or asm**

config mode [options] <modename>

## azure feature: commands to manage account features

**List all features available for your subscription**

feature list [options]

**Shows a feature**

feature show [options] <providerName> <featureName>

**Registers a previewed feature of a resource provider**

feature register [options] <providerName> <featureName>

## azure group: Commands to manage your resource groups

**Creates a new resource group**

group create [options] <name> <location>

**Set tags to a resource group**

group set [options] <name> <tags>

**Deletes a resource group**

group delete [options] <name>

**Lists the resource groups for your subscription**

group list [options]

**Shows a resource group for your subscription**

group show [options] <name>

**Commands to manage resource group logs**

group log show [options] [name]

**Commands to manage your deployment in a resource group**

group deployment create [options] [resource-group] [name]

group deployment list [options] <resource-group> [state]

group deployment show [options] <resource-group> [deployment-name]

group deployment stop [options] <resource-group> [deployment-name]

## azure location: Commands to get the available locations for all resource types

**List the available locations**

location list [options]

## azure network: Commands to manage network resources

**Commands to manage virtual networks**

To see all commands : azure network -h

network vnet create [options] <resource-group> <name> <location>

Allows to create a new virtual network. In the following example we create a virtual network named newvnet for resource group myresourcegroup in the West US region.

azure network vnet create myresourcegroup newvnet "west us"

info: Executing command network vnet create

+ Looking up virtual network "newvnet"

+ Creating virtual network "newvnet"

Loading virtual network state

data: Id: /subscriptions/###############################/resourceGroups/myresourcegroup/providers/Microsoft.Network/virtualNetworks/newvnet

data: Name: newvnet

data: Type: Microsoft.Network/virtualNetworks

data: Location: westus

data: Tags:

data: Provisioning state: Succeeded

data: Address prefixes:

data: 10.0.0.0/8

data: DNS servers:

data: Subnets:

data:

info: network vnet create command OK

Parameter options:

-h, --help output usage information

-v, --verbose use verbose output

--json use json output

-g, --resource-group <resource-group> the name of the resource group

-n, --name <name> the name of the virtual network

-l, --location <location> the location

-a, --address-prefixes <address-prefixes> the comma separated list of address prefixes for this virtual network

For example -a 10.0.0.0/24,10.0.1.0/24.

Default value is 10.0.0.0/8

-d, --dns-servers <dns-servers> the comma separated list of DNS servers IP addresses

-t, --tags <tags> the tags set on this virtual network.

Can be multiple. In the format of "name=value".

Name is required and value is optional.

For example, -t tag1=value1;tag2

-s, --subscription <subscription> the subscription identifier

network vnet set [options] <resource-group> <name>

Updates a virtual network configuration within a resource group.

azure network vnet set myresourcegroup newvnet

info: Executing command network vnet set

+ Looking up virtual network "newvnet"

+ Updating virtual network "newvnet"

+ Loading virtual network state

data: Id: /subscriptions/###############################/resourceGroups/myresourcegroup/providers/Microsoft.Network/virtualNetworks/newvnet

data: Name: newvnet

data: Type: Microsoft.Network/virtualNetworks

data: Location: westus

data: Tags:

data: Provisioning state: Succeeded

data: Address prefixes:

data: 10.0.0.0/8

data: DNS servers:

data: Subnets:

data:

info: network vnet set command OK

Parameter options:

-h, --help output usage information

-v, --verbose use verbose output

--json use json output

-g, --resource-group <resource-group> the name of the resource group

-n, --name <name> the name of the virtual network

-a, --address-prefixes <address-prefixes> the comma separated list of address prefixes for this virtual network.

For example -a 10.0.0.0/24,10.0.1.0/24.

This list will be appended to the current list of address prefixes.

The address prefixes in this list should not overlap between them.

The address prefixes in this list should not overlap with existing address prefixes in the vnet.

-d, --dns-servers [dns-servers] the comma separated list of DNS servers IP addresses.

This list will be appended to the current list of DNS server IP addresses.

-t, --tags <tags> the tags set on this virtual network.

Can be multiple. In the format of "name=value".

Name is required and value is optional. For example, -t tag1=value1;tag2.

This list will be appended to the current list of tags

--no-tags remove all existing tags

-s, --subscription <subscription> the subscription identifier

network vnet list [options] <resource-group>

The command allows to list all virtual networks in a resource group.

C:\>azure network vnet list myresourcegroup

info: Executing command network vnet list

+ Listing virtual networks

data: ID

Name Location Address prefixes DNS servers

data: -------------------------------------------------------------------

------ -------- -------- ---------------- -----------

data: /subscriptions/###############################/resourceGroups/

wvnet newvnet westus 10.0.0.0/8

info: network vnet list command OK

Parameter options:

-h, --help output usage information

-v, --verbose use verbose output

--json use json output

-g, --resource-group <resource-group> the name of the resource group

-s, --subscription <subscription> the subscription identifier

The command removes a virtual network.

azure network vnet delete myresourcegroup newvnetX

info: Executing command network vnet delete

+ Looking up virtual network "newvnetX"

Delete virtual network newvnetX? [y/n] y

+ Deleting virtual network "newvnetX"

info: network vnet delete command OK

Parameter options:

-h, --help output usage information

-v, --verbose use verbose output

--json use json output

-g, --resource-group <resource-group> the name of the resource group

-n, --name <name> the name of the virtual network

-q, --quiet quiet mode, do not ask for delete confirmation

-s, --subscription <subscription> the subscription identifier

**Commands to manage virtual network subnets**

network vnet subnet create [options] <resource-group> <vnet-name> <name>

command allows to add another subnet to an existing virtual network.

azure network vnet subnet create -g myresourcegroup --vnet-name newvnet -n subnet --address-prefix 10.0.1.0/24

network vnet subnet set [options] <resource-group> <vnet-name> <name>

Sets a specific virtual network subnet within a resource group.

azure network vnet subnet set -g myresourcegroup --vnet-name newvnet -n subnet1

azure network vnet subnet list [options] <resource-group> <vnet-name>

Lists all the virtual network subnets for a specific virtual network within a resource group.

azure network vnet subnet set -g myresourcegroup --vnet-name newvnet -n subnet1

network vnet subnet show [options] <resource-group> <vnet-name> <name>

Displays virtual network subnet properties

azure network vnet subnet show -g myresourcegroup --vnet-name newvnet -n subnet1

## azure provider: Commands to manage resource provider registrations

**List currently registered providers in ARM**

provider list [options]

**Show details about the requested provider namespace**

provider show [options] <namespace>

**Register provider with the subscription**

provider register [options] <namespace>

**Unregister provider with the subscription**

provider unregister [options] <namespace>

## azure storage: Commands to manage your Storage objects

**Commands to manage your Storage accounts**

storage account list [options]

storage account show [options] <name>

storage account create [options] <name>

storage account set [options] <name>

storage account delete [options] <name>

**Commands to manage your Storage account keys**

storage account keys list [options] <name>

storage account keys renew [options] <name>

**Commands to show your Storage connection string**

storage account connectionstring show [options] <name>

**Commands to manage your Storage blobs**

storage blob list [options] [container] [prefix]

storage blob show [options] [container] [blob]

storage blob delete [options] [container] [blob]

storage blob upload [options] [file] [container] [blob]

storage blob download [options] [container] [blob] [destination]

**Commands to manage your blob copy operations**

storage blob copy start [options] [sourceUri] [destContainer]

storage blob copy show [options] [container] [blob]

storage blob copy stop [options] [container] [blob] [copyid]

**Commands to manage shared access signature of your Storage blob**

storage blob sas create [options] [container] [blob] [permissions] [expiry]

**Commands to manage your Storage file shares**

storage share create [options] [share]

storage share show [options] [share]

storage share delete [options] [share]

storage share list [options] [prefix]

**Commands to manage your Storage files**

storage file list [options] [share] [path]

storage file delete [options] [share] [path]

storage file upload [options] [source] [share] [path]

storage file download [options] [share] [path] [destination]

**Commands to manage your Storage file directory**

storage directory create [options] [share] [path]

storage directory delete [options] [share] [path]

**Commands to manage your Storage queues**

storage queue create [options] [queue]

storage queue list [options] [prefix]

storage queue show [options] [queue]

storage queue delete [options] [queue]

**Commands to manage shared access signatures of your Storage queue**

storage queue sas create [options] [queue] [permissions] [expiry]

**Commands to manage stored access policies of your Storage queue**

storage queue policy create [options] [queue] [name]

storage queue policy show [options] [queue] [name]

storage queue policy list [options] [queue]

storage queue policy set [options] [queue] [name]

storage queue policy delete [options] [queue] [name]

**Commands to manage your Storage logging properties**

storage logging show [options]

storage logging set [options]

**Commands to manage your Storage metrics properties**

storage metrics show [options]

storage metrics set [options]

**Commands to manage your Storage tables**

storage table create [options] [table]

storage table list [options] [prefix]

storage table show [options] [table]

storage table delete [options] [table]

**Commands to manage shared access signatures of your Storage table**

storage table sas create [options] [table] [permissions] [expiry]

**Commands to manage stored access policies of your Storage table**

storage table policy create [options] [table] [name]

storage table policy show [options] [table] [name]

storage table policy list [options] [table]

storage table policy set [options] [table] [name]

storage table policy delete [options] [table] [name]

## azure tag: Commands to manage your resource manager tag

**Add a tag**

tag create [options] <name> <value>

**Remove an entire tag or a tag value**

tag delete [options] <name> <value>

**Lists the tag information**

tag list [options]

**Get a tag**

tag show [options] [name]

## azure vm: Commands to manage your Azure Virtual Machines

**Create a VM**

vm create [options] <resource-group> <name> <location> <os-type>

**Create a VM with default resources**

vm quick-create [options] <resource-group> <name> <location> <os-type> <image-urn> <admin-username> <admin-password

[AZURE.TIP]Starting with CLI version 0.10, you can provide a short alias such as "UbuntuLTS" or "Win2012R2Datacenter" as the image-urn for some popular Marketplace images. Run azure help vm quick-create for options. Additionally, starting with version 0.10, azure vm quick-create uses premium storage by default if it's available in the selected region.

**List the virtual machines within an account**

vm list [options]

**Get one virtual machine within a resource group**

vm show [options] <resource-group> <name>

**Delete one virtual machine within a resource group**

vm delete [options] <resource-group> <name>

**Shutdown one virtual machine within a resource group**

vm stop [options] <resource-group> <name>

**Restart one virtual machine within a resource group**

vm restart [options] <resource-group> <name>

**Start one virtual machine within a resource group**

vm start [options] <resource-group> <name>

**Shutdown one virtual machine within a resource group and releases the compute resources**

vm deallocate [options] <resource-group> <name>

**List available virtual machine sizes**

vm sizes [options]

**Capture the VM as OS Image or VM Image**

vm capture [options] <resource-group> <name> <vhd-name-prefix>

**Set the state of the VM to Generalized**

vm generalize [options] <resource-group> <name>

**Get instance view of the VM**

vm get-instance-view [options] <resource-group> <name>

**Enable you to reset Remote Desktop Access or SSH settings on a Virtual Machine and to reset the password for the account that has administrator or sudo authority**

vm reset-access [options] <resource-group> <name>

**Update VM with new data**

vm set [options] <resource-group> <name>

**Commands to manage your Virtual Machine data disks**

vm disk attach-new [options] <resource-group> <vm-name> <size-in-gb> [vhd-name]

vm disk detach [options] <resource-group> <vm-name> <lun>

vm disk attach [options] <resource-group> <vm-name> [vhd-url]

**Commands to manage VM resource extensions**

vm extension set [options] <resource-group> <vm-name> <name> <publisher-name> <version>

vm extension get [options] <resource-group> <vm-name>

**Commands to manage your Docker Virtual Machine**

vm docker create [options] <resource-group> <name> <location> <os-type>

**Commands to manage VM images**

vm image list-publishers [options] <location>

vm image list-offers [options] <location> <publisher>

vm image list-skus [options] <location> <publisher> <offer>

vm image list [options] <location> <publisher> [offer] [sku]

# Using Azure CLI Commands to create a Linux VM

There are multiple ways to create environmens in Microsoft Azure. In this exercise we will look at two different ones; creating a vm using quick-create and after that using the create command for more granularity and control.

### Method 1: Using the Azure CLI command Azure VM Create:

Using the azure vm create gives you more control on parameters than using quick-create. With this command you can specify networks, subnets, storage accounts etc at the time of deployment.

First delete the resource group created in the previous exercises to start with a new resource group. Important: all resources you previously created will be lost.

View all resource groups in your subscription

azure group list

Then delete the specific resource group:

azure group delete [resource group name]

Create a new storage group using the last 4 digits of your mobile phone number as a unique identifier

azure group create accelerate[uniqueid] -l westeurope

Verify the RG using the JSON parser

azure group show accelerate[unique] --json | jq '.'

Create the Storage Account

azure storage account create -g Accelerate[UNIQUEID] -l westeurope --type GRS accelerate[UNIQUEID]storage

Verify the storage using the JSON parser

azure storage account show -g accelerate[UNIQUEID] accelerate[UNIQUEID]storage --json | jq '.'

Create the Virtual Network

azure network vnet create -g accelerate[UNIQUEID] -n acceleratevnet -a 192.168.0.0/16 -l westeurope

Create the Subnet

azure network vnet subnet create -g accelerate[UNIQUEID] -e acceleratevnet -n FrontEnd -a 192.168.1.0/24

Verify the VNet and Subnet using the JSON parser

azure network vnet show accelerate[UNIQUEID] acceleratevnet --json | jq '.'

Create a public IP

azure network public-ip create -g accelerate[UNIQUEID] -n acceleratelbpip -l westeurope -d acceleratelb -a static -i 4

Create the load balancer

azure network lb create -g accelerate[UNIQUEID] -n acceleratelb -l westeurope

Create a front-end IP pool for the load balancer, and associate our public IP

azure network lb frontend-ip create -g accelerate[UNIQUEID] -l acceleratelb -n acceleratefrontendpool -i acceleratelbpib

Create our back-end IP pool for the load balancer

azure network lb address-pool create -g accelerate[UNIQUEID] -l Acceleratelb -n Acceleratefrontendpool

Create SSH inbound NAT rules for the load balancer

azure network lb inbound-nat-rule create -g Accelerate[UNIQUEID] -l Acceleratelb -n VM1-SSH -p tcp -f 4222 -b 22

azure network lb inbound-nat-rule create -g Accelerate[UNIQUEID] -l Acceleratelb -n VM2-SSH -p tcp -f 4223 -b 22

Create our web inbound NAT rules for the load balancer

azure network lb rule create -g Accelerate[UNIQUEID] -l Acceleratelb -n WebRule -p tcp -f 80 -b 80 \

-t Acceleratefrontendpool -o Acceleratefrontendpool

Create our load balancer health probe

azure network lb probe create -g Accelerate[UNIQUEID] -l Acceleratelb -n HealthProbe -p "http" -f healthprobe.aspx -i 15 -c 4

Verify the load balancer, IP pools, and NAT rules using the JSON parser

azure network lb show -g Accelerate[UNIQUEID] -n Acceleratelb --json | jq '.'

Create the first NIC

azure network nic create -g Accelerate[UNIQUEID] -n LB-NIC1 -l westeurope --subnet-vnet-name Acceleratevnet --subnet-name FrontEnd

Create the second NIC

azure network nic create -g Accelerate[UNIQUEID] -n LB-NIC2 -l westeurope --subnet-vnet-name Acceleratevnet --subnet-name FrontEnd

Verify the NICs using the JSON parser

azure network nic show Accelerate[UNIQUEID] LB-NIC1 --json | jq '.'

azure network nic show Accelerate[UNIQUEID] LB-NIC2 --json | jq '.'

Create the NSG

azure network nsg create -g Accelerate[UNIQUEID] -n AccelerateNSG -l westeurope

Add the inbound rules for the NSG

azure network nsg rule create --protocol tcp --direction inbound --priority 1000 --destination-port-range 22 --access allow -g Accelerate[UNIQUEID] -a AccelerateNSG -n SSHRule

azure network nsg rule create --protocol tcp --direction inbound --priority 1001 --destination-port-range 80 --access allow -g Accelerate[UNIQUEID] -a AccelerateNSG -n HTTPRule

Verify the NSG and inbound rules using the JSON parser

azure network nsg show -g Accelerate[UNIQUEID] -n AccelerateNSG --json | jq '.'

Bind the NSG to the NICs

azure network nic set -g Accelerate[UNIQUEID] -n LB-NIC1 -o AccelerateNSG

azure network nic set -g Accelerate[UNIQUEID] -n LB-NIC2 -o AccelerateNSG

Create the availability set

azure availset create -g Accelerate[UNIQUEID] -n AccelerateAvailSet -l westeurope

Create the first Linux VM (type as one command)

azure vm create \

--resource-group Accelerate[UNIQUEID] \

--name LinuxVM1 \

--location westeurope \

--os-type linux \

--availset-name AccelerateAvailSet \

--nic-name LB-NIC1 \

--vnet-name Acceleratevnet \

--vnet-subnet-name FrontEnd \

--storage-account-name Accelerate[UNIQUEID]storage \

--image-urn suse:openSUSE:13.2:2015.05.12 \

--admin-username ops

To shorten the command, you can also use the options listed when you type in azure help vm create.

Create the second Linux VM

azure vm create \

--resource-group Accelerate[UNIQUEID] \

--name LinuxVM2 \

--location westeurope \

--os-type linux \

--availset-name AccelerateAvailSet \

--nic-name LB-NIC1 \

--vnet-name Acceleratevnet \

--vnet-subnet-name FrontEnd \

--storage-account-name Accelerate[UNIQUEID]storage \

--image-urn suse:openSUSE:13.2:2015.05.12 \

--admin-username ops

Verify everything built using the JSON parser

azure vm show -g Accelerate[UNIQUEID] -n LinuxVM1 --json | jq '.'

azure vm show -g Accelerate[UNIQUEID] -n LinuxVM2 --json | jq '.'

### Method 2: Using the Azure CLI command Azure VM Quick-Create:

As you noticed, manually typing all commands to create a working solution, while fully customizable, takes quite some time and effort. For quicker deployments, you can use the azure quick-create command.

First delete the resource group created in the previous exercises. Important: all resources you previously created will be lost.

View all resource groups in your subscription

azure group list

Then delete the specific resource group:

azure group delete [resource group name]

Create a new storage group using the last 4 digits of your mobile phone number as a unique identifier

azure group create accelerate[uniqueid] -l westeurope

From a terminal or command prompt, run the following command to list all the available locations from which you can choose to create a virtual machine. Take note of one of them (e.g.: *WestEurope*); as you will use it in the following step.

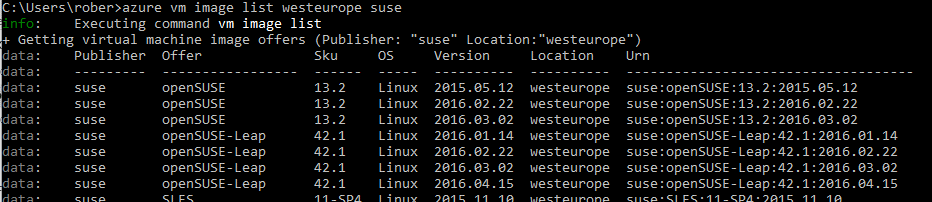
azure location list

Select the Linux image you would like to deploy. For this example we choose OpenSuse 13.2

Get a list of all available images :

azure vm image list westeurope suse

Note the Image URN



For a select number of images in the Azure Marketplace, aliases are available. So you can use CentOS, Debian, OpenSuse, RHEL for example as the URN

To create a new virtual machine based on this image run the following command. Replace the values in [ ] with your specific values for the virtual machine name and administrator user.

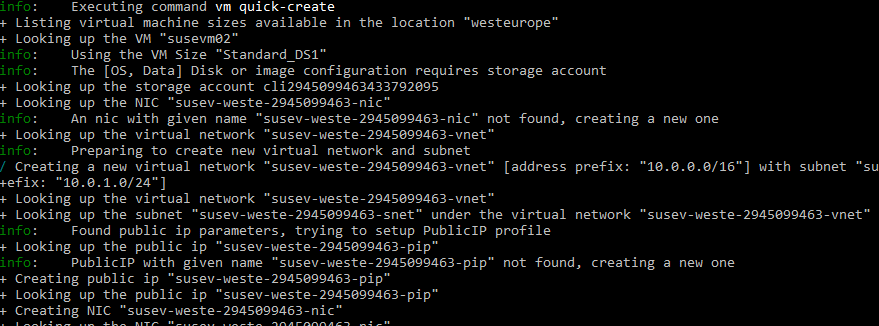
As the command executes it will prompt you to enter the password for the admin user.

azure vm quick-create -g [RESOURCEGROUP] -n [VMNAME] -l [LOCATION] -y [OSTYPE] -Q [URN] [USER]

example: azure vm quick-create -g accelerate1574 -n susevm02 -l westeurope -y linux -Q opensuse arobakk

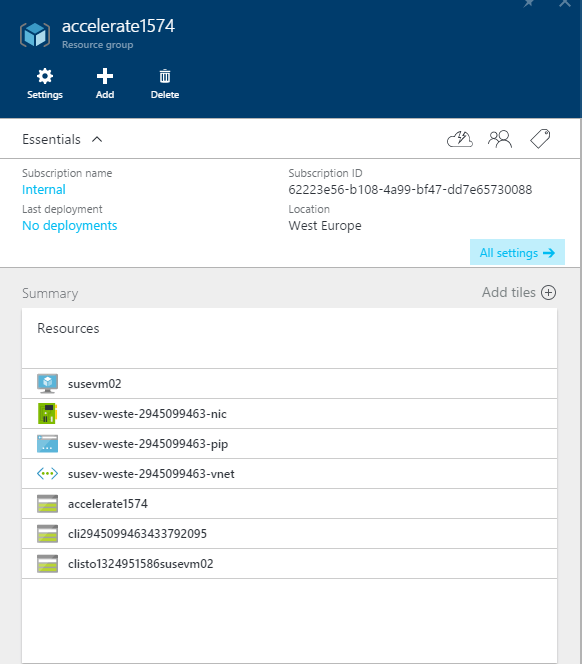
For more detailed help, type azure help vm quick-create

You should see output like:



Note that the storage account, virtual networks etc. are all automatically populated.

When you go to <https://portal.azure.com> and check the resource group you just created, you should see something similar like this image:



# Installing and using GIT as a repository

How to upload web site to **Microsoft Azure** using Visual Studio Code and GitHub public/private repository.

Requirements for this tutorials:

* DreamSpark, MSDN or BizSpark subscription, or any business account
* Visual Studio Code
* Git source control installed on your PC/Mac/Linux machine

... and good luck with this tutorial!

## Install NODE.JS

### Installing Node.js

Download install package from [https://nodejs.org](https://nodejs.org/). At the time of writing this, we have tested and validated the sample application against Node 5.5.0 and npm 3.3.12. Download the specific release of Node 5.5.0 from [here](https://nodejs.org/en/download/releases/) to ensure later modules in this training work properly.

When you install Node.js, you’ll want to ensure your PATH variable includes your install path so you can call Node from anywhere. Node comes with npm installed so you should have a version of npm. To install a specific version of npm globallly, run the following:

npm install -g npm@3.3.12

### Testing Your Install

Create a new directory named hello-world, add a new app.js file with the following content:

/\* app.js \*/

console.log('Hello World!');

In the command prompt (or terminal on Mac) , run

$ node app.js

If you get any error regarding Node is not found, open a new command prompt (or terminal) to reflect the new environment variable for Node.

To stop the application, run Ctrl+C.

### Installing and Setting up Git

Chances are, you already have git running on your computer or you at least have heard of Git before. For this training, you will need to setup git locally and know few basics about git.

### Installing Git

If you don’t have Git installed, navigate to the official [Git Download page and download it](https://git-scm.com/download).

#### Getting Git for Mac

There are several ways to install Git on a Mac. You can choose from one of the following ways:

1) If you want a more up to date version, you can also install it via a binary installer. An OSX Git installer is maintained and available for download at the Git website, at <http://git-scm.com/download/mac>.

2) If you already have Homebrew, you can install Git by executing:

$ brew install git

Make sure you update your $PATH environment variable to include the latest install path of Git. For example:

**$ echo 'export PATH="/usr/local/bin:/usr/local/sbin:~/bin:$PATH"' >> ~/.bash\_profile**

#### Getting Git for Windows

If you’re already using Chocolatey or Windows 10’s package manager to install software, you can simply run the following command from an elevated Powershell or Bash console (right click, select ‘Run as Administrator’):

$ cinst git.install

$ cinst poshgit

# Restart PowerShell / CMDer before moving on - or run

$ env:Path = [System.Environment]::GetEnvironmentVariable("Path","Machine") + ";" + [System.Environment]::GetEnvironmentVariable("Path","User")

$ cinst git-credential-winstore

$ cinst github

#### Verifying Git Installation

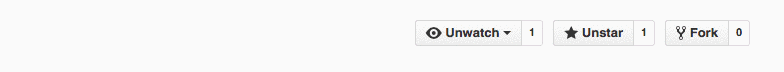
Now that you have Git installed, open up PowerShell on Windows or terminal on Mac. If everything worked correctly, you should be able to run git --version.

### Signing up for a Free GitHub account

Before we can get started, you need to register with GitHub for a free account. Either create or login into your account on [GitHub](https://github.com/join).

### Forking a Repository to your Account

Now you are ready to do more with Git. Let’s start with a sample project. Head over to the Accelerate Lab repository on <https://github.com/robert-bakker/AccelerateLab> and click the little ‘fork’ button in the upper right.

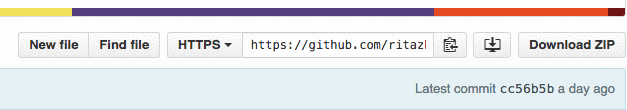


Fork repo to your own

This will create a copy of the repository as it exists in the original account into your own account.

### Cloning the Repository to Your Machine

Visit your fork (which should be at github.com/{your\_github\_username}/acceleratelab) and copy the “HTTPS Clone URL”. Using this URL, you’re able to clone the repository, which downloads the whole repository, including its history and information about its origin locally. From PowerShell on Windows or terminal on Mac, change into the directory where you would like to clone your repo.

Copy your repo URL

Clone the code to your local machine.

$ git clone https://github.com/{your\_github\_username}/acceleratelab

This should generate output that looks roughly like this:

$ git clone https://github.com/{your\_github\_username}/acceleratelab

Cloning into 'acceleratelab'...

...

...

Checking connectivity... done.

You can run explorer . from PowerShell on Windows or open . from Terminal on Mac to open up the folder in Explorer or Finder respectively. All the files are there - including the history of the whole repository. The connection to your fork ({your\_github\_username}/acceleratelab) is still there.

### Creating a new Branch for your repo

In modern Git development, every single change that you want to make to the code base will be made in a “branch”. Like a tree branch, the branch is “based” on a different branch, and unlike other SCM systems, Git branches are very lightweight. In our case, your base branch isgh-pages. The default branch name for GitHub repositories is master. In order to create a new branch, you can always run:

# This makes sure that your new branch is based on master

$ git checkout gh-pages

# When the default branch of a repo is "master", you should "git checout master" instead

# The below command creates a new branch

$ git checkout -b my-new-branch

You can now go ahead and make your changes - adding files, writing code, fixing bugs. Keep in mind that a branch should host isolated changes. For example, you should create one branch that fixes a bug, then another branch for to develop a new feature you want to implement.

### Staging your Changes for a Commit

Now that you made your changes, you can “stage” them for a commit. Whenever you stage a file for a commit, you make a snapshot of the file at the time you’re staging it for a commit. If you change a file after you staged it, you will have to stage it again. To stage a file, simply run:

$ git add ./path-to/my-file.md

If you just want to stage all files in your current repository (including deletions), run:

$ git add --all .

### Committing your Changes

Now that your changes have been staged, we’re ready to commit them. You can either pass the commit command a title for your commit - or omit the parameter, in which case Git will open up the default text editor for you to create a commit message.

To commit the quick way:

$ git commit -m "Add new feature: Git is Awesome"

To commit the long way, allowing you to define both title and message of your commit:

$ git commit

### Pushing your new Branch to Your Fork on GitHub

Let’s say you have implemented a new feature, made some changes, committed the changes - now we have to make sure that your changes also end up on GitHub. To do so, we have to push your local branch to your fork on GitHub. Run the command below, using the name of the branch you want to push to

$ git checkout NAME\_OF\_YOUR\_NEW\_BRANCH

$ git push -u origin NAME\_OF\_YOUR\_NEW\_BRANCH

# Using ARM templates to further automate VM Deployments

## Use the Azure CLI for Mac, Linux, and Windows with Azure Resource Manager

### Azure resources

Use the Azure Resource Manager to create and manage a group of resources (user-managed entities such as a virtual machine, database server, database, or website) as a single logical unit, or resource group.

One advantage of the Azure Resource Manager is that you can create your Azure resources in a declarative way: you describe the structure and relationships of a deployable group of resources in JSON templates. The template identifies parameters that can be filled in either inline when running a command or stored in a separate JavaScript Object Notation (JSON) parameters file. This allows you to easily create new resources using the same template by simply providing different parameters. For example, a template that creates a website will have parameters for the site name, the region the website will be located in, and other common settings.

When a template is used to modify or create a group, a deployment is created, which is then applied to the group.

After you create a deployment, you can manage the individual resources imperatively on the command line, just like you do in the classic deployment model. For example, use CLI commands in Resource Manager mode to start, stop, or delete resources such as [Azure Resource Manager virtual machines](https://github.com/Azure/azure-content/blob/master/articles/virtual-machines/virtual-machines-linux-cli-deploy-templates.md). https://github.com/Azure/azure-content/blob/master/articles/virtual-machines/virtual-machines-linux-cli-deploy-templates.md

## Authentication

Working with the Azure Resource Manager through the Azure CLI currently requires you to authenticate to Microsoft Azure by using the azure login command and then specifying an account managed by Azure Active Directory - either a work or school account (an organizational account) or a Microsoft account.

## Set the Resource Manager mode

Because the CLI is not in Resource Manager mode by default, use the following command to enable Azure CLI Resource Manager commands.

azure config mode arm

## Create a resource group

A resource group is a logical grouping of resources such as network, storage, and compute resources. Almost all commands in the Resource Manager mode need a resource group.

First delete the resource group created in the previous exercises. Important: all resources you previously created will be lost.

View all resource groups in your subscription

azure group list

Then delete the specific resource group:

azure group delete [resource group name]

Create a new storage group using the last 4 digits of your mobile phone number as a unique identifier

azure group create -n accelerate[uniqueid] -l westeurope

You will deploy to this accelerate[ID] resource group later when you use a template to launch a Linux VM. Once you have created a resource group, you can add resources like virtual machines and networks or storage.

## Use resource group templates

When working with templates, you can either [create your own](https://github.com/Azure/azure-content/blob/master/articles/resource-group-authoring-templates.md) (https://github.com/Azure/azure-content/blob/master/articles/resource-group-authoring-templates.md), or use one of the community-contributed [Quick Start templates](https://azure.microsoft.com/documentation/templates/),( https://azure.microsoft.com/documentation/templates/) which are also available on [GitHub](https://github.com/Azure/azure-quickstart-templates) (https://github.com/Azure/azure-quickstart-templates).

### Creating a 5 node VMM scale set running an Apache/PHP website.

To create this more advanced scenario (VM Scale Set, loadbalancer, Apache & PHP), using an azure resource template from github we will use the [201-vmss-lapstack-autoscale](https://github.com/Azure/azure-quickstart-templates/tree/master/201-vmss-lapstack-autoscale) template created by Madhan Arumugam Ramakrishnan, available on the Azure QuickStart templates repository on GitHub. <https://github.com/Azure/azure-quickstart-templates/blob/master/201-vmss-lapstack-autoscale/>

This template creates a Simple self-contained Ubuntu/Apache/PHP (LAP stack) with auto scaling & load balancing. The VM Scale Set scales up when avg CPU across all VMs is larger than 60%, and it automatically scales down again when the average CPU is less than 50%. This template will deploy the following:

* Deploy the VM Scale Set with an instance count of 1
* After it is deployed look at the resource group public IP address resource (in portal or resources explorer). Get the IP or domain name.
* Browse to the website (port 80), which shows the current backend VM name.
* Hit the "Do work" button with an iteration count of say 300 (represents seconds of max CPU).
* After a few minutes the VM Scale Set capacity will increase, and refreshing the browser and going to the home page a few times will show additional backend VM name(s).
* You can increase the work by connecting to more backend websites, or decrease by letting the iterations time-out, in which case the VM Scale Set will scale down - hence after about 10 minutes the capacity should be back down to 1.

### Deploy this template to your Azure resource group

You can use this template directly from [GitHub](https://github.com/Azure/azure-quickstart-templates), instead of downloading one to your computer. To do this, pass the URL to the azuredeploy.json file for the template in your command by using the **--**template**-uri** option. To get the URL, open azuredeploy.json on GitHub in raw mode, and copy the URL that appears in the browser's address bar. You can then use this URL directly to create a deployment by using a command similar to the following. (the new resource group needs to be created first using: azure group create -n accelerate[uniqueid] -l westeurope)

azure group deployment create [Resource Group Name] [Resource Group Deployment Name] --template-uri [RAW Github link to azuredeploy.json]

azure group deployment create accelerateLAP accelerateLAPdeployment https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/201-vmss-lapstack-autoscale/azuredeploy.json

You are then prompted to enter the necessary template parameters.

vmssName : the name of the VM Scale Set

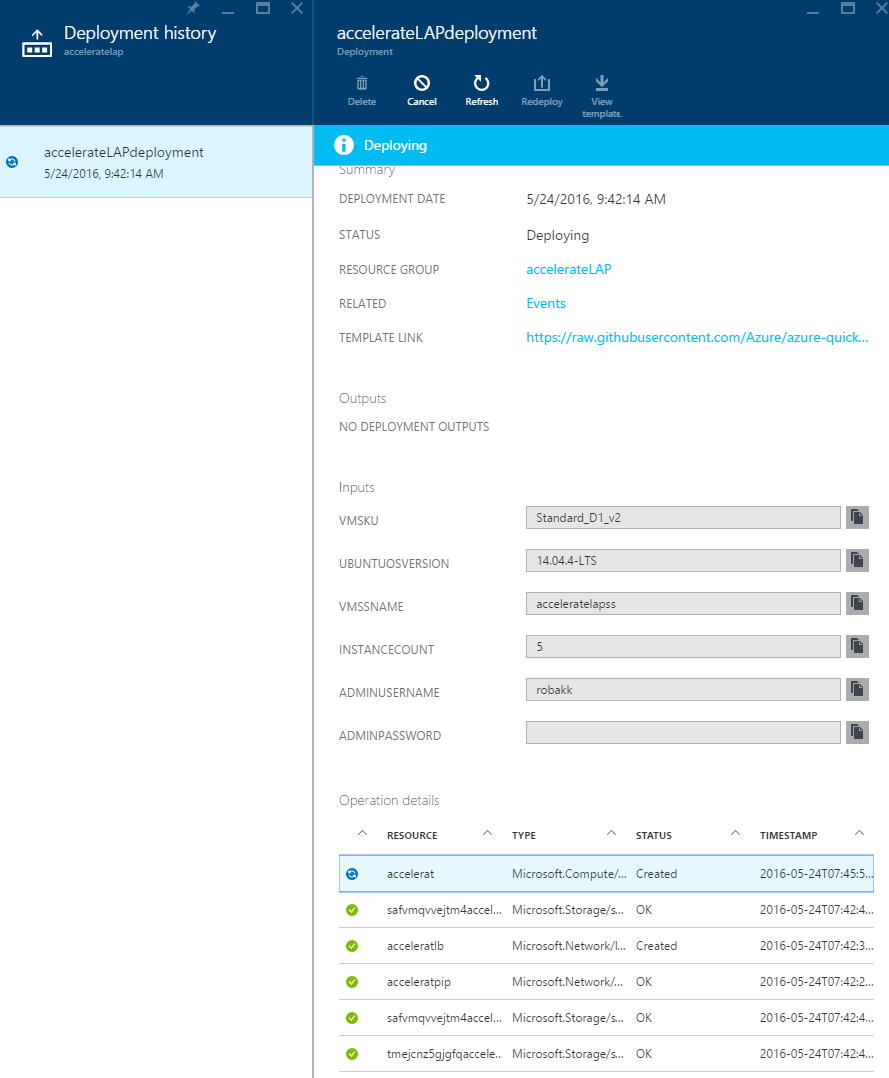
instancecount : how many VMs need to be in the scale set

adminusername : user name for the administrator

Password: the admin password

This will take approximately 5 minutes. You can watch the progress of the deployment on portal.azure.com in the resource group specified:

Now you should be able to use a web browser to connect to [scalesetname].westeurope.cloudapp.azure.com and enter a number of seconds to



## Export a resource template

For an existing resource group, you can view the Resource Manager template for the resource group. Exporting the template offers two benefits:

1. You can easily automate future deployments of the solution because all of the infrastructure is defined in the template.
2. You can become familiar with template syntax by looking at the JSON that represents your solution.

Using the Azure CLI, you can either export a template that represents the current state of your resource group, or download the template that was used for a particular deployment.

* **Export the template for a resource group** - This is helpful when you have made changes to a resource group, and need to retrieve the JSON representation of its current state. However, the generated template contains only a minimal number of parameters and no variables. Most of the values in the template are hard-coded. Before deploying the generated template, you may wish to convert more of the values into parameters so you can customize the deployment for different environments.

To export the template for a resource group to a local directory, run the azure group export command as shown in the following example. (Substitute a local directory appropriate for your operating system environment.)

azure group export accelerate[uniqueid] ~/azure/templates/

* **Download the template for a particular deployment** -- This is helpful when you need to view the actual template that was used to deploy resources. The template will include all of the parameters and variables defined for the original deployment. However, if someone in your organization has made changes to the resource group outside of what is defined in the template, this template will not represent the current state of the resource group.

To download the template used for a particular deployment to a local directory, run the azure group deployment template download command.

azure group deployment template download TestRG testRGDeploy ~/azure/templates/downloads/

Template export is in preview, and not all resource types currently support exporting a template. When attempting to export a template, you may see an error that states some resources were not exported. If needed, you can manually define these resources in your template after downloading it.

To view and edit these JSON templates, Visual Studio Code or Visual Studio are recommended. Install visual studio code for Windows, OSX or Linux from here : <https://code.visualstudio.com/>