

Virtual Networks

Monday, September 25, 2017 5:42 PM

Build an enterprise-grade, secure virtual network in Microsoft Azure and monitor its performance.

This workshop is intended to help users become familiar with networking in Microsoft Azure. Participants will create a virtual network and examine various ways to manage and interact with the network and its resources.

What You Will Learn

- Virtual Networks
- Resource Groups
- Automation through ARM Templates
- Managing NICs
- Managing IP Addresses
- Azure Event Logs

Ideal Audience

- CISOs and VPs of Information Security
- CIOs
- IT Managers
- Active Directory and Network Administrators

Overview

This workshop is intended to help users become familiar with networking in Microsoft Azure. Participants will create a virtual network and examine various ways to manage and interact with the network and its resources.

Time Estimate: x.x hours

Requirements

Setup Requirements

The following workshop assumes that you have used the Azure Workshops CLI to pre-create the necessary lab environment. To use the Azure Workshops CLI, you will need the following applications installed on your local machine:

- [Node.js](#)
- [Git](#)

As stated above, these tools are necessary for downloading and running the CLI locally. Download and install these tools according to the instructions on their respective website.

Additional Requirements

Additionally, you will need a subscription (trial or paid) to both Office 365 and Microsoft Azure. Please see the [next](#) page for how to create trial subscriptions in both.

Office 365 and Azure Registration

Demo Domain

For the purposes of this workshop, you will need a demo domain name - a domain name that you will *not* be required to register with a domain name registrar (DNR), but will be used as your fictitious company. We, of course, do not want to use any domain names associated with production accounts.

The simple way to do this is allow a service to create one for us. So, to create a random domain name, we'll actually use a random username generator.

Open a browser to <http://jimpix.co.uk/words/random-username-generator.asp> and click the green "Go!" button close to the top of the page. Upon doing this, you will be presented with 25 different two-word combinations. Pick one that you like or click the green "Refresh" button until you do.

Once you find a domain name, write it down; you will use it for the remainder of the workshop.

Office 365

Now that we have a domain name, let's create a 1-month trial Office 365 account. This will automatically create a domain in Azure AD which we'll connect to virtual datacenter later in the workshop.

Direct your browser to <https://products.office.com/en-us/business/office-365-affiliate-program-try-business-premium>. In order to take advantage of some of the Azure Active Directory premium features, we will need the Business Premium edition of Office 365.

1. Begin by clicking on the green button "Start your free business trial".
2. Complete the form on the first page:
 - Choose your country (this cannot be changed later due to data sovereignty and other factors)
 - Enter your name
 - Enter an email address (this should be a *legitimate email address* as this will be the administrator's security/reset email)
 - Enter a phone number (enter a *legitimate cell phone number* in order to test multi-factor authentication)
 - Enter your company name from above
 - Choose a company size

3. For the form on the second page:

- Enter a username for yourself in a format you prefer (e.g. if your name was John Doe, you could enter: john.doe, jdoe, john_doe, etc.)
- For your company, enter the company name from above (NOTE: you will see here that the initial domain name will be *yourcompany.onmicrosoft.com*. This is the Azure Active Directory domain to which we will connect later in the workshop.) If your domain name has already been used, try another one from the previous list.
- Enter and confirm your password

4. Prove you are not a robot by entering a telephone number at which you can receive a text or phone call.

5. Enter the code that was text'ed to you or that you received from the auto-attendant.

It should take less than a minute to create your account. After the process is complete, you should see a message stating that you are ready to go. While your account was *created* in less than a minute, it may take up to another 15 minutes or so to finish creating all of the additional services in Office 365. That's fine, as it will be a while before we actually need them.

Finally, remember this trial account is only good for 30 days. While Microsoft will not initially *delete* your account, they will disable functionality.

Azure

Finally, we need to create a trial Azure subscription. Believe it or not, we are already using Azure Active Directory because we just set up Office 365. Office 365 uses Azure AD underneath to manage all of our exchange users. We simply need to create a subscription so that we can leverage Azure's other offerings.

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

IMPORTANT: On the sign-up form page, you should see your new email address that associated with your new Office 365 account. If not, click on **Sign Out** and re-authenticate using your newly formed credentials (e.g. *username@yourcompany.onmicrosoft.com*).

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

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

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

Setup

Installing the CLI

Once you have the requisites installed, you will then need to install the CLI. The CLI can be installed from the command-line or terminal prompt using Node.js.

First, open a command-line window or terminal prompt. Then, type the following command:

```
npm install azworkshops-cli -g
```

Running this command will take a few seconds to complete. But, doing so will download the Azure Workshops CLI, along with its dependencies, into a directory that is located in a globally accessible path.

Azure Subscription

As stated in the requirements section, the workshop requires an active Azure subscription.

It is recommended that you do not use an Azure subscription that is currently being used for production. The CLI will create it's own resource groups, but it is not the best practice to utilize production environments for testing and workshops, such as this.

For best results, it is recommended that you setup register for the trial subscription as outlined on the [previous](#) page.

Creating the Lab Environment

The automated building of the lab environment can take approximately 30 minutes to complete. It is best to begin this process while you are reviewing the workshop material.

Verify Installation of the CLI

From a prompt, enter the following command:

```
azworkshops --version
```

A successful execution of the command should print the current version of the Azure Workshops CLI which can be found in the right column, slightly down the page, of the Node Package Manager [website](#). If you do not see a version number, return to the requirements [setup](#) and try reinstalling them.

If you successfully see the correct version number, you are ready to begin the lab setup.

Build the Environment

From a prompt, enter the following:

```
azworkshops
```

1. You will be presented with a menu from which to choose a base configuration. Choose the base configuration for **Basic Active Directory**.
2. You will then need to authenticate with Azure. Visit <http://aka.ms/devicelogin> and enter the code provided to you.
3. Choose the subscription that you would like to use for this workshop.
4. Select the location for the created resources. It is best to choose a location that is closest to you in order to reduce latency.
5. You will then be prompted with additional configuration questions.
 1. For the AD domain name, enter your company name from the previous page with '.local' as the TLD (e.g. mycompany.local).
 2. For the NETBIOS name, it should automatically be an ALLCAPS version of the company name that you just entered (without the '.local' TLD extension). If so, just press Enter to accept the default. If not, enter a valid NETBIOS name.
6. After completing the configuration questions, the building of the lab environment will begin. Once completed, you will be presented with all of the lab's configured settings (e.g. resource group, domain, domain admin, password, etc.) It is best to copy this down for future use.



Exploring Azure

Objective

The first objective is simply for you to become familiar with connecting to and navigating the Azure portal.

Azure Portal Basics

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

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

This is where we will get started creating our resources.




Creating a Resource Group

Objective

In this module, we will create our first resource group and examine some of its properties along with viewing what *type* of resources we're allowed to add. We'll discuss some basic compliance around resource groups and determining where the metadata is stored. We will also add a basic resource to the group to view how the resources are listed, managed, etc. Finally, we will complete this module by deleting the resource group along with all of its resources.

Create a Resource Group

As stated previously, in order to create resources in Azure, we must assign them to a resource group. So let's get started by creating our first resource group.

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

NOTE: When you create a resource group, you are prompted to choose a location. Additionally, as you create individual resources, you will also be prompted to choose locations. The location of resource groups and their resources can be *different*. This is because resource groups store *metadata* describing their contained resources; and, due to some types of compliance that your company may

adhere to, you may need to store that metadata in a different location than the resources themselves. For example, if you are a US-based company, you may choose to keep the metadata state-side while creating resources in foreign regions to reduce latency.

Overview

Overview

Before we jump in to building our virtual network, let's take a little bit of time exploring Resource Groups. In this section, we're going to explore some of the predominant features of Resource Groups.

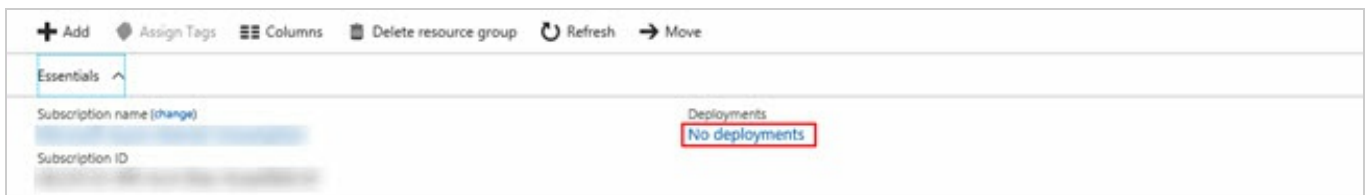
Note

Much of the following sections will initially be empty in Azure as we've not created any resources in our Resource Group. That's okay; we'll come back to them later. The next few pages exist to simply provide an introduction.

Once we've created our Resource Group, or whenever we select a Resource Group in Azure, the first blade we are directed to is the **Overview**. The Overview blade does just that - gives us an overview of the resources that our Resource Group contains. Here, we can add resources, delete resources and move resources to other groups or subscriptions.

Deployments

One thing to notice on this blade is **Deployments**. Every time we add or remove a set of resources in batch, this is considered a "deployment".



What is a batch? If we were to simply add a single network card to our resource group, that would be considered a batch or *deployment*. On the other hand, deploying a virtual machine, which would not only be the VM, itself, but it could also include a network interface (NIC), an IP address, a storage account, etc. The process of deploying a VM could potential require adding a few resources in a single deployment.

As we deploy more and more resources to our Resource Group, we can see the deployment history. This is useful for, among other reasons, auditing and change management.


We will come back to this blade later to add our Virtual Network.

Activity Log

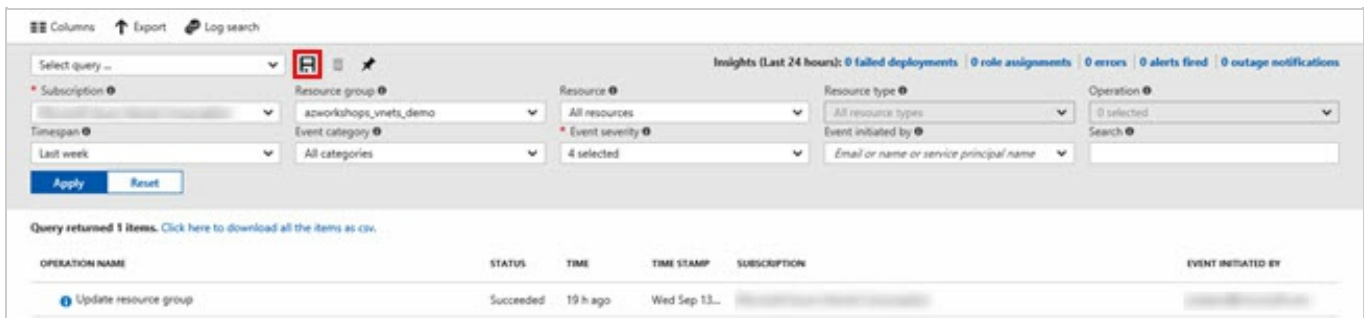
Overview

Let's take some time to explore how to query the history and events within our Resource Group.

Activity Log

Once you've created a Resource Group or you click on a Resource Group, the first tab **Overview** is selected. Of course, there's no resources created yet. We'll create some momentarily. For now, in the left panel, click on **Activity log** .

The Activity Log gives you a report of all events within your Resource Group. At the top of the query window, you'll see various options to customize your query. Additionally, you can choose to save frequent search parameters.



The screenshot shows the Azure Activity Log interface. At the top, there are tabs for 'Columns', 'Export', and 'Log search'. Below these are various filters: 'Subscription' (set to 'azworkshops_vnets_demo'), 'Resource group' (set to 'azworkshops_vnets_demo'), 'Resource' (set to 'All resources'), 'Resource type' (set to 'All resource types'), 'Event category' (set to 'All categories'), 'Event severity' (set to '4 selected'), and 'Operation' (set to '0 selected'). There are also buttons for 'Apply' and 'Reset'. Below the filters, it says 'Query returned 1 items. Click here to download all the items as csv.' Below this is a table with the following columns: 'OPERATION NAME', 'STATUS', 'TIME', 'TIME STAMP', 'SUBSCRIPTION', and 'EVENT INITIATED BY'. The table contains one row: 'Update resource group', 'Succeeded', '19 h ago', 'Wed Sep 13...', 'Subscription: azworkshops_vnets_demo', and 'Event initiated by: [redacted]'.

OPERATION NAME	STATUS	TIME	TIME STAMP	SUBSCRIPTION	EVENT INITIATED BY
Update resource group	Succeeded	19 h ago	Wed Sep 13...	Subscription: azworkshops_vnets_demo	Event initiated by: [redacted]

There's, of course, not much to see here at this time. But, as we complete this workshop, you can revisit this blade periodically to see the changes to your Resource Group.

Exploring the Log

By default, the query looks at all events in the past 6 hours. You should see an operation entitled *Update resource group* referring to the creation of the Resource Group. If you don't see this event, try increasing the **Timespan** and click **Apply**.

Once you see the event listed, go ahead and click on the first event, **Update resource group**.

Clicking on the event will reveal an information window at the bottom of the screen. The information window initially shows you the same basic information that the log table showed. However, there are two other features exposed by this window.

+ Add activity log alert

SummaryJSON

Operation name
Update resource group

Time stamp
Wed Sep 13 2017 21:58:14 GMT-0400 (Eastern Daylight Time)

Event initiated by
joshua@msn.com

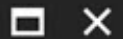
Log Details

Clicking on the **JSON** "tab" will display the details of the log entry in a JSON format. Looking through this JSON object, you can find a vast amount of different information regarding the event. Such information includes the action performed, the person performing the action, originating IP address, log level, resource type(s), resource location(s), and subscription Id. All of this information can be used to create automated alerts in Azure.

Alerts

The second feature of the details window enables you to create alerts around specific events. In the details window, clicking on **Add activity log alert** will open a new blade to create an alert.

Add activity log alert



* Activity log alert name ⓘ

Description ⓘ

* Subscription ⓘ

* Resource group ⓘ

Source

Input ⓘ Activity log

Criteria

* Event category ⓘ

Resource type ⓘ

Resource group ⓘ

Resource ⓘ

Operation name ⓘ

Level ⓘ

Status ⓘ

Event initiated by ⓘ

Alert via

Action group ☒ New ☐ Existing

* Action group name ⓘ

* Short name ⓘ

Actions

NAME

ACTION TYPE

DETAILS

The name of the receiver

SMS



Depending on the action. ...



Note that only the country code '1' is currently supported for SMS.

It can take up to 5 minutes for an Activity log alert to become active.

OK

Let's create an informational alert.

In the alert blade, enter the following information:

1. Basics

- Activity log alert name: **VNet Info Alert**
- Description: **Informational alert for resource group**
- Subscription: **<choose your subscription>**
- Resource group: **<accept the default>** (this will create a new resource group for alerts)

2. Source

- Input: **<Activity log>**

3. Criteria

- Event category: **Administrative**
- Resource type: **All**
- Resource group: **<azworkshops_vnets_demo>**
- Resource: **All**
- Operation name: **All**
- Level: **All**
- Status: **All**
- Event initiated by: **<accept the default>**

4. Alert via

- Action group: **New**
- Action group name: **Information Alerts**
- Short name: **info** (include the greater and less than signs)

5. Actions

Here, add two actions - one for your email address and another for your mobile phone (SMS).

IMPORTANT: The *name of the receiver* must be **unique**. Therefore, for your email address and SMS, name the receiver with a differentiator (e.g. "John Doe - Email", "John Doe - SMS").

Notice that you can also specify a *Webhook* (URL) if you want to automate a notification or ticket in your support application, such as *ServiceNow*.

6. Click **OK**.

It should only take a couple of seconds to create the Resource Group (if necessary) and create/update the Action Group. Once the process has completed you should receive a text (SMS) message and an email - provided that you set up both actions - alerting you that you've been added to the Action Group.

Powershell/CLI

Before we move forward in looking at the other options of our Resource Group, let's interact with the Activity Logs just a bit more. This time we'll query the logs using Powershell and/or the CLI. As a reminder, most people use Powershell in a Windows environment and the CLI in Linux-based environment, but either can be used on any environment.

Both of the following sections are the same. Simply use the environment with which you feel most comfortable.

Powershell

1. From your prompt connect to Azure:

```
login-azurermaccount
```

2. Make sure you are "attached" to the correct subscription. If you only have one subscription, it will automatically be chosen for you. However, if you have more than one subscription, then the incorrect one may have been chosen by default.

1. List all subscriptions:

```
get-azurermsubscription | select -Property SubscriptionName
```

2. Set the active subscription:

```
select-azurermsubscription -SubscriptionName "<your subscription name>"
```

3. Let's get *all* of the log entries for our Resource Group:

```
get-azurermlog -ResourceGroup azworkshops_vnets_demo
```

You should see a couple of entries here, if nothing else, regarding the creation of the Resource Group.

4. Now, let's get *all* of the log entries where the *Status* is **succeeded**:

```
get-azurermlog -ResourceGroup azworkshops_vnets_demo -Status Succeeded
```

This is helpful when you want to query any log entries where an operation may have **failed**.

5. You can also specify a date range and a person who performed the action:

```
get-azurermlog -ResourceGroup azworkshops_vnets_demo -StartTime (Get-Date
).AddDays(-14) -Caller <your email address>
```

This queries all actions performed in the last two weeks by you. If you want to specify a right boundary on your time frame, then you can use the `EndTime` parameter.

6. Finally, to display all of the JSON-formatted details that you saw earlier in the details window, add the `DetailedOutput` parameter:

```
Get-AzureRmLog -ResourceGroup azworkshops_vnets_demo -Status Succeeded -D
etailedOutput
```

CLI

1. From your prompt connect to Azure:

```
azure login
```

You will need to visit <https://aka.ms/devicelogin> and enter the code that the CLI gives you.

2. Make sure you are "attached" to the correct subscription. If you only have one subscription, it will automatically be chosen for you. However, if you have more than one subscription, then the incorrect one may have been chosen by default.

1. List all subscriptions:

```
azure account list | awk -F' {2,}' '{print $2}'
```

(NOTE: You can simply issue the `azure account list` command to show all subscriptions *and* other details. However, in a Linux-based environment, the above command will strip out all of the other information and show the subscription names only.)

2. Set the active subscription:

```
azure account set "<your subscription name>"
```

3. Let's get *all* of the log entries for our Resource Group:

```
azure group log show azworkshops_vnets_demo -a
```

You should see a couple of entries here, if nothing else, regarding the creation of the Resource Group.

(NOTE: The `-a` parameter for "all" entries is optional. However, remember from above that, by default, only the past 6 hours are queried from the log. So, if you want anything beyond that, you'll need to specify this parameter.)

- Now, let's get *all* of the log entries where the *Status* is **succeeded**. The CLI doesn't have a querying option, so we'll need to convert the output to JSON then filter that output using `jq`. The *Status* is a property of the objects returned from the Azure Insight SDK. The *Status* property has a property of its own called *Value*. For accessing these properties and filter by them, we'll use 'dot' notation:

```
azure group log show azworkshops_vnets_demo -a --json | jq '[] | select(.status.value == "Succeeded")'
```

The command takes all of our returned data and, using `jq`, converts it to a JSON array of objects, then filters and displays only objects whose property *Status.Value* equals "Succeeded". This is helpful when you want to query any log entries where an operation may have **failed**.

- You can also filter by date range and a person who performed the action:

```
azure group log show azworkshops_vnets_demo -a --json | jq --arg startTime $(date --date="14 days ago" -Is) '[] | select(.eventTimestamp > $startTime and .caller == "<your email address>")'
```

This queries all actions performed in the last two weeks by you. Looking at this, what we're doing is passing an argument `startTime` in to `jq`. The value of `startTime` is calculated by running a *nested* command `$(date --date="14 days ago" -Is)`. That value is then used by `jq` to compare to the `eventTimestamp` property of our returned log entries.

If you want to specify a right boundary on your time frame, then you can create an additional argument and condition.

(NOTE: Mac's do not support the `--date` or `-I` parameter. Instead, your nested command should be `$(date -v-14d +%Y-%m-%dT%H:%M:%S%z)` where the `-v` subtracts 14 days and the output is formatted.)

Wow! That was a lot of interaction with our Activity Logs. However, understanding how to use Activity Logs is *extremely* beneficial for many reasons. Those logs can later be audited, piped, or used for reporting in platforms such as Operations Management Suite (OMS). While this may have seemed

like a lot of work, it will get easier over time and it's highly recommended that you continue to practice with the logs and become familiar with them.

Access Control (IAM)

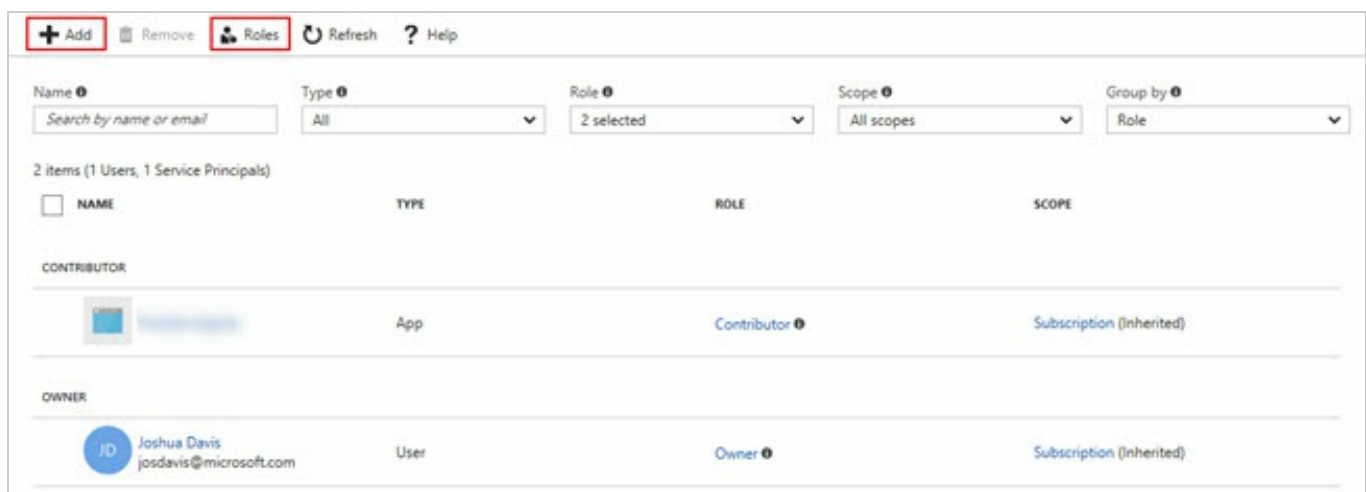
Overview

We're briefly going to discuss how to secure our group and its resources by utilizing Azure's RBAC capabilities.

Access Control (IAM)

The Access Control blade is the management interface enabling you to leverage Azure's Identity and Access Management (IAM). Azure's IAM is used to secure your cloud resources, whether that's a group, as a whole, or individual resources. Azure's IAM follows the common permissions model of Role Based Access Control (RBAC) which is a method of controlling access to resources based on the roles of the individual users within the enterprise.

By default, this view shows all explicitly assigned roles/users who have access to this resource. In our case, this is for our resource group with which we're currently working (e.g. *azworkshops_vnets_demo*).



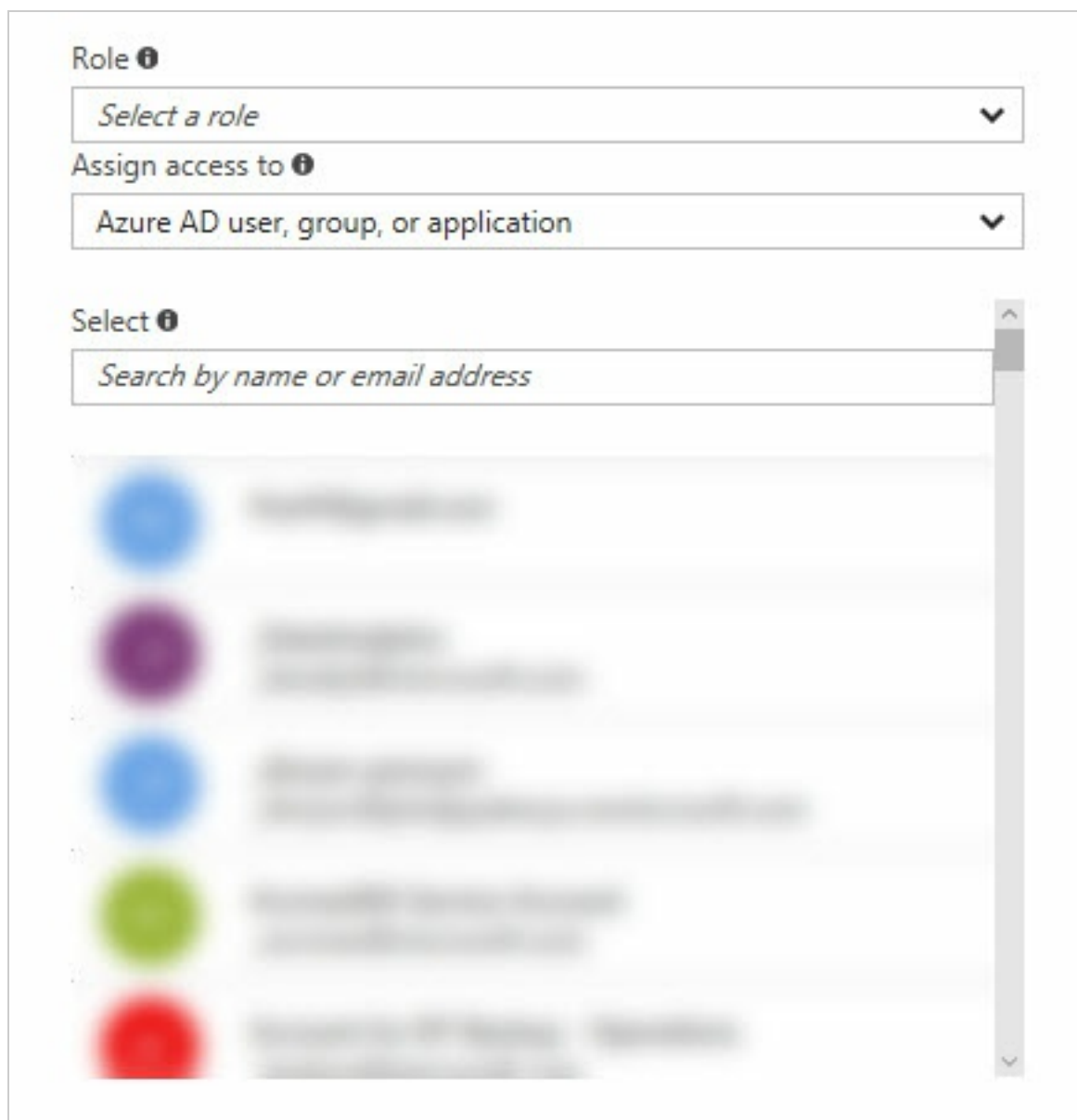
In the sample image, you'll see that I have two types of users with assigned roles to my resource group. In the far most right corner, you'll see that these two users were *inherited* from the subscription level. Therefore, we know that these two users were added to a policy at the subscription and those roles have been "passed down" to the child resources of the subscription.

The first user is a *Contributor* while the second user is an *Owner*. A Contributor pretty much has full rights to a resource, but cannot grant/revoke access. An Owner, on the other hand, can do whatever they please, including management of access.

Finally, in the second column we see the user's *type*. The second user is a standard user account in Active Directory. The first user, on the other hand, is described as an *app*. What that means is that our first user is a registered application, also known as a *Service Principal*. This registered application has been assigned a Contributor role to all of the resources within our subscription. Therefore, the application itself can interact and manage our subscription.

Adding Users

At the top of the blade, you will see **Add**. Clicking on this will reveal something similar to the following configuration blade.



The image shows a configuration blade for assigning roles in Azure. It contains three main sections:

- Role**: A dropdown menu with the placeholder text "Select a role".
- Assign access to**: A dropdown menu with the placeholder text "Azure AD user, group, or application".
- Select**: A search bar with the placeholder text "Search by name or email address". Below the search bar is a list of search results, each consisting of a colored circular icon and a text label. The results are partially obscured by a vertical scrollbar on the right.

By expanding the dropdown for *Role*, you'll see that there are over 60 different roles available for assignment to users and/or groups. Each role allows the user(s) to perform specific tasks within Azure and its resources. If you are interested in learning about each specific role, checkout the [Azure](#)

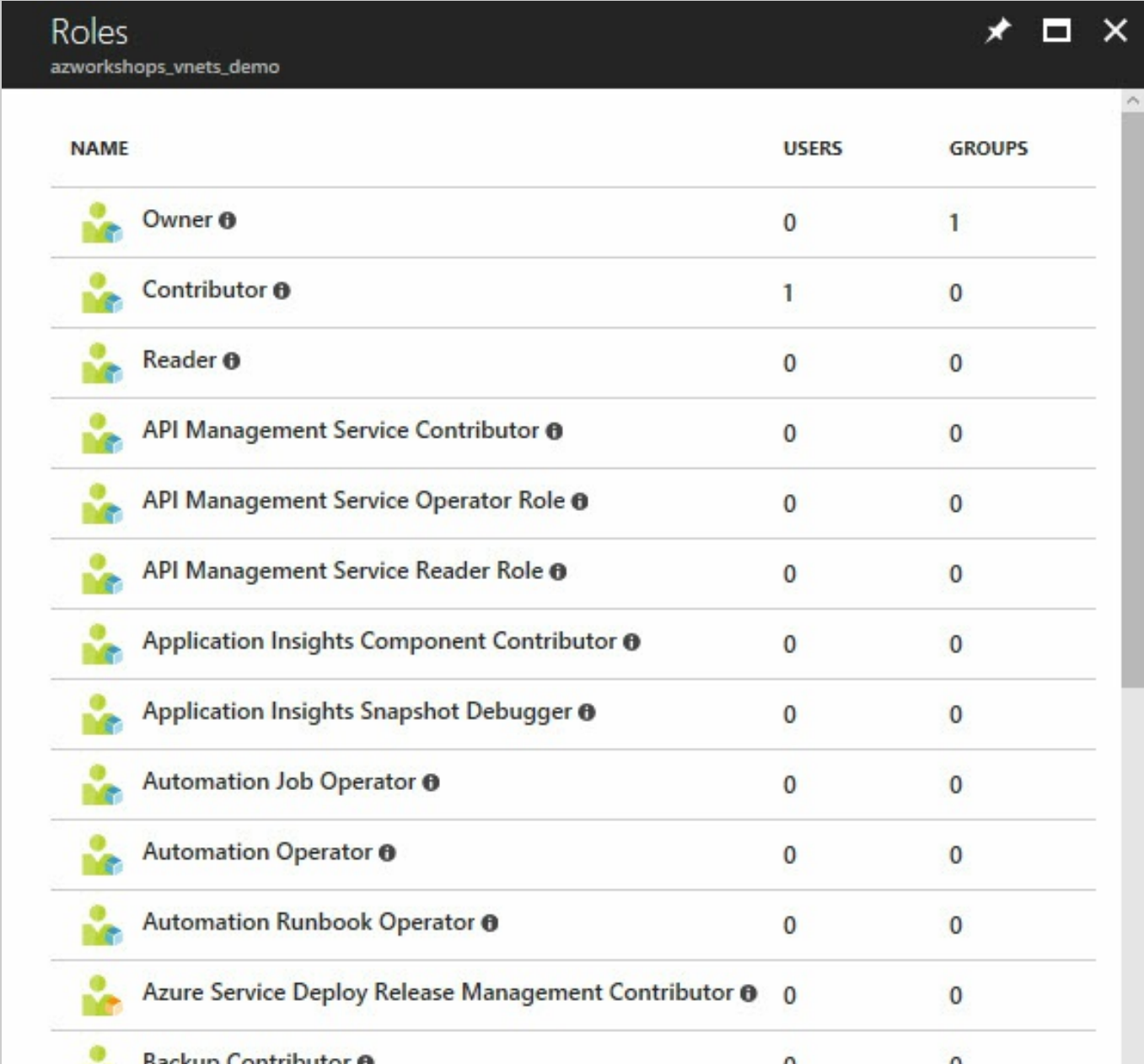
[docs](#).

You have the ability to decide whether you desire to grant access to an Azure AD user, group, or application; or, you may grant access to a virtual machine.














If you choose a virtual machine, a few other settings will appear allowing you to choose the virtual machine. If not, you can simply choose a current user in Active Directory, a Service Principal, a Service Account, or enter someone's email address to give access outside of your Active Directory.

Viewing Roles

Azure also provides a way to very easily see what users and/or groups have been assigned to what roles. Returning to the main **Access Control (IAM)** blade and clicking on **Roles** will display a table of all the roles along with a count of how many users and groups have been assigned to that role.



The screenshot shows the 'Roles' blade in the Azure portal. The title bar includes the word 'Roles' and the subscription name 'azworkshops_vnets_demo'. The table lists various roles with their respective user and group counts.

NAME	USERS	GROUPS
 Owner ⓘ	0	1
 Contributor ⓘ	1	0
 Reader ⓘ	0	0
 API Management Service Contributor ⓘ	0	0
 API Management Service Operator Role ⓘ	0	0
 API Management Service Reader Role ⓘ	0	0
 Application Insights Component Contributor ⓘ	0	0
 Application Insights Snapshot Debugger ⓘ	0	0
 Automation Job Operator ⓘ	0	0
 Automation Operator ⓘ	0	0
 Automation Runbook Operator ⓘ	0	0
 Azure Service Deploy Release Management Contributor ⓘ	0	0
 Backup Contributor ⓘ	0	0

Clicking on any one of these roles will display the assigned users and groups to that particular role.

Powershell/CLI

We can use Powershell and the CLI to work with RBAC, as well. I won't get into *assigning* or *removing* users from roles in this workshop as we don't have any other users. But, we will experiment with querying our RBAC policies. Just as we did on the previous page, we will practice using Powershell and the CLI. Based on your environment and needs, you can choose which tool to use as both of the following sections are the same.

Powershell

1. From your prompt connect to Azure:

```
login-azurermaccount
```

2. Make sure you are "attached" to the correct subscription. If you only have one subscription, it will automatically be chosen for you. However, if you have more than one subscription, then the incorrect one may have been chosen by default.

1. List all subscriptions:

```
get-azurermsubscription | select -Property SubscriptionName
```

2. Set the active subscription:

```
select-azurermsubscription -SubscriptionName "<your subscription name>"
```

3. Let's list RBAC roles that are available for assignment and inspect the operations to which they grant access:

```
get-azurermroledefinition | FT Name, Description
```

4. Let's list the available actions for a given role:

```
get-azurermroledefinition "Contributor" | FL Actions, NotActions  
  
(get-azurermroledefinition "Virtual Machine Contributor").Actions
```

The first command shows all of the actions and "not-actions" (e.g. allow and deny) for the *Contributor* role. The second command shows what actions someone who has the *Virtual Machine Contributor* is allowed to perform.

5. Show role assignments for the resource group:

```
get-azureroleassignment -ResourceGroupName azworkshops_vnets_demo | FT DisplayName, RoleDefinitionName, Scope
```

6. Show roles assigned to a specific user:

```
get-azureroleassignment -SignInName <your email address> | FT DisplayName, RoleDefinition, Scope  
  
get-azureroleassignment -SignInName <your email address> -ExpandPrincipalGroups | FT DisplayName, RoleDefinition, Scope
```

The first command shows all roles assigned to the user *explicitly*. The second command shows all roles *inherited* by the user due to their membership in group.

CLI

1. From your prompt connect to Azure:

```
azure login
```

You will need to visit <https://aka.ms/devicelogin> and enter the code that the CLI gives you.

2. Make sure you are "attached" to the correct subscription. If you only have one subscription, it will automatically be chosen for you. However, if you have more than one subscription, then the incorrect one may have been chosen by default.

1. List all subscriptions:

```
azure account list | awk -F' {2,}' '{print $2}'
```

(NOTE: You can simply issue the `azure account list` command to show all subscriptions *and* other details. However, in a Linux-based environment, the above command will strip out all of the other information and show the subscription names only.)

2. Set the active subscription:

```
azure account set "<your subscription name>"
```

3. Let's list RBAC roles that are available for assignment and inspect the operations to which they grant access:

```
azure role list --json | jq '.[ ] | {"name":.Name, "description":.Description}'
```

4. Let's list the available actions for a given role:

```
azure role show "contributor" --json | jq '.[ ] | {"Actions":.Actions,"NotActions":.NotActions}'
```

```
azure role show "virtual machine contributor" --json | jq '.[ ] | .Actions'
```

The first command shows all of the actions and "not-actions" (e.g. allow and deny) for the *Contributor* role. The second command shows what actions someone who has the *Virtual Machine Contributor* is allowed to perform.

5. Show role assignments for the resource group:

```
azure role assignment list --resource-group azworkshops_vnets_demo --json | jq '.[ ] | {"DisplayName":.properties.aADObject.displayName, "RoleName":.properties.roleName, "Scope":.properties.scope}'
```

6. Show roles assigned to a specific user:

```
azure role assignment list --signInName <your email address> --json | jq '.[ ] | {"DisplayName":.properties.aADObject.displayName, "RoleDefinitionName":.properties.roleName, "Scope":.properties.scope}'
```

```
azure role assignment list --signInName <your email address> --expandPrincipalGroups --json | jq '.[ ] | {"DisplayName":.properties.aADObject.displayName, "RoleDefinitionName":.properties.roleName, "Scope":.properties.scope}'
```

The first command shows all roles assigned to the user *explicitly*. The second command shows all roles *inherited* by the user due to their membership in group.

This introduction to RBAC should be sufficient in managing access to your Azure resources. Again, you can perform a lot more management via Powershell or the CLI, but for the purposes of this workshop, the Azure GUI will work just fine.

Tags

Resource Costs

Deployments

Policies

Automation Script

Metrics

Alert Rules

Diagnostic Logs

Resource Providers

Adding Resources

Managing Resources

Automation