Module 2: Restrict access to data in Azure

**Overview**

In this module, you learn how to restrict access to your data both from external public endpoints, but also other services internal to Azure. You will use PowerShell to configure virtual network service endpoints for your Azure SQL Data Warehouse and Data Lake Storage account.

**Pre-requisites:**

* Owner permissions on an Azure subscription
* Existing Azure SQL Data Warehouse
* Existing Azure Data Lake Storage (Gen2) instance
* Existing Azure virtual network
* Completed pre-requisites from Module 0

**Login to your Azure subscription in PowerShell**

The rest of this module will use PowerShell to configure settings and values. If you haven’t previously logged into your Azure account, run the following to setup your PowerShell session.

# ------- Edit the variables below to set session-wide variables ---------

$subscriptionName = '<SubscriptionName>'

$participantNumber = <ParticipantNumber>

$resourceGroupName = '<ResourceGroupName>'

# Log into your Azure account

Connect-AzAccount -Subscription $SubscriptionName

Run the following to setup the variables you’ll use to configure Azure PowerShell commands in the lab:

# --------------- Setup module variables ----------------------------------

$serverName = 'usgsserver' + $participantNumber

$dataLakeAccountName = 'usgsdatalake' + $participantNumber

$dataWarehouseName = 'usgsdataset'

$virtualNetworkName = 'usgsvirtualnetwork' + $participantNumber

$subNetName = 'usgsSubnet'

$dbUserName = 'usgsadmin';

$dbPassword = 'P@ssword' + $participantNumber

$securePasswordString = ConvertTo-SecureString $dbPassword -AsPlainText -Force

$dbCredentials = New-Object System.Management.Automation.PSCredential($dbUserName, $securePasswordString)

**Restrict global access to Azure Data Lake Storage instance**

By default, your Azure Data Lake Storage instance is setup to accept connections from any client on any network. You can setup network rules to restrict access to specific networks and Azure services. These rules give you fine-grained control over which Azure services and applications can access the raw data stored in the lake. In this step, you will configure the data lake to restrict access to just trusted Azure services.

In your existing PowerShell session

1. Configure the data lake to deny connections by default

# Display the default rule for the data lake

(Get-AzStorageAccountNetworkRuleSet -ResourceGroupName $resourceGroupName -AccountName $dataLakeAccountName).DefaultAction

# Configure network rules to deny access by default

Update-AzStorageAccountNetworkRuleSet -ResourceGroupName $resourceGroupName -Name $dataLakeAccountName -DefaultAction Deny

1. Add an exception that allows connections from trusted Microsoft services

# Configure network rules to allow access from trusted Microsoft services and logging pipelines

Update-AzStorageAccountNetworkRuleSet -ResourceGroupName $ResourceGroupName -Name $DataLakeAccountName -Bypass AzureServices,Metrics,Logging

**Create managed-service identity for your Azure SQL Data Warehouse**

Now that access to your data lake is restricted to just trusted Azure services, you need to create a trusted identity for your Azure SQL Data Warehouse instance. This identity will be used to authenticate to the lake and enables data import and export scenarios using Polybase technology.   
  
In your existing PowerShell session:

1. Register your Azure SQL Server instance with Azure Active Directory.

# Generate and assign an Azure AD Identity for this server

Set-AzSqlServer -ResourceGroupName $resourceGroupName -ServerName $serverName -AssignIdentity

1. Grant your Azure SQL Server access to the data lake instance. The SQL Server will be assigned to the role of Storage Blob Data Contributor.  
   Note: Only members with the Owner RBAC privilege on the data lake can perform this step.

# List details of the Storage Blob Data Contributor Role

$StorageContributorRole = "Storage Blob Data Contributor (Preview)"

Get-AzRoleDefinition -Name $StorageContributorRole

# Get ServicePrincipalId assigned to SQL Server

$serverAzureAdIdentity = (Get-AzADServicePrincipal -SearchString $ServerName).Id

# Grant server access to data lake (requires Owner permissions)

$subscriptionId = (Get-AzSubscription -SubscriptionName $subscriptionName).SubscriptionId

$permissionScope = "/subscriptions/$subscriptionId/resourceGroups/$resourceGroupName/providers/Microsoft.Storage/storageAccounts/$dataLakeAccountName"

New-AzRoleAssignment -ObjectId $serverAzureAdIdentity -RoleDefinitionName $StorageContributorRole -Scope $permissionScope

1. Create an external table in Azure SQL DataWarehouse to confirm access to the data lake instance. You should see a message that reads the sample file upload to the data lake during the lab initialization.

# Create external table to read test message in data lake

$databaseCredentialQuery = "CREATE MASTER KEY; CREATE DATABASE SCOPED CREDENTIAL readme\_cred WITH IDENTITY = 'Managed Service Identity';"

$externalDataSourceQuery = "CREATE EXTERNAL DATA SOURCE [usgsdatalakereadme] WITH (TYPE=HADOOP, LOCATION=N'abfss://readme@$dataLakeAccountName.dfs.core.windows.net', CREDENTIAL = readme\_cred)"

$externalFileFormatQuery = "CREATE EXTERNAL FILE FORMAT [usgsdatalakereadmeformat] WITH (FORMAT\_TYPE = DELIMITEDTEXT, FORMAT\_OPTIONS (FIELD\_TERMINATOR = N'|', USE\_TYPE\_DEFAULT = False))"

$externalTableQuery = "CREATE EXTERNAL TABLE [staging].[dataLakeAccessTest]([header] [nvarchar](40) NOT NULL,[notice] [nvarchar](150) NOT NULL)

WITH (DATA\_SOURCE = [usgsdatalakereadme],LOCATION = N'/readme.txt',FILE\_FORMAT = [usgsdatalakereadmeformat],REJECT\_TYPE = VALUE,REJECT\_VALUE = 0)"

Invoke-Sqlcmd -Query "CREATE schema staging" -ServerInstance $fullyQualifiedServerName -Database $dataWarehouseName -Username $dbCredentials.UserName -Password $dbCredentials.GetNetworkCredential().Password

Invoke-Sqlcmd -Query $databaseCredentialQuery -ServerInstance $fullyQualifiedServerName -Database $dataWarehouseName -Username $dbCredentials.UserName -Password $dbCredentials.GetNetworkCredential().Password

Invoke-Sqlcmd -Query $externalDataSourceQuery -ServerInstance $fullyQualifiedServerName -Database $dataWarehouseName -Username $dbCredentials.UserName -Password $dbCredentials.GetNetworkCredential().Password

Invoke-Sqlcmd -Query $externalFileFormatQuery -ServerInstance $fullyQualifiedServerName -Database $dataWarehouseName -Username $dbCredentials.UserName -Password $dbCredentials.GetNetworkCredential().Password

Invoke-Sqlcmd -Query $externalTableQuery -ServerInstance $fullyQualifiedServerName -Database $dataWarehouseName -Username $dbCredentials.UserName -Password $dbCredentials.GetNetworkCredential().Password

# Issue data warehouse query to test access to data lake storage instance.

Invoke-Sqlcmd -Query "SELECT \* FROM staging.dataLakeAccessTest" -ServerInstance $fullyQualifiedServerName -Database $dataWarehouseName -Username $dbCredentials.UserName -Password $dbCredentials.GetNetworkCredential().Password

**Add Azure Data Lake Storage to Virtual Network**  
In this section, you will configure your data lake instance to allow access only from a specific virtual network. This will be done by configuring an Azure Data Lake Storage service endpoint within the existing virtual network. This endpoint gives traffic an optimal route directly to the data lake instance.   
  
Note: to apply a virtual network rule to a storage account, the user must have ‘Join Service to a subnet’ permission. This is typically included in the Storage Account Contributor built-in role.

In your existing PowerShell session

1. Enable service endpoint for Azure Data Lake Storage on an existing virtual network and subnet

# Get existing virtual network

$vnetInstance = Get-AzVirtualNetwork -ResourceGroupName $resourceGroupName -Name $virtualNetworkName

# Configure virtual network goal state

$subnetConfig = Set-AzVirtualNetworkSubnetConfig -Name $subNetName -VirtualNetwork $vnetInstance -AddressPrefix "10.0.0.0/24" -ServiceEndpoint "Microsoft.Storage"

# Enable service endpoint for Azure storage service on virtual network subnet

Set-AzVirtualNetwork -VirtualNetwork $subnetConfig

1. Add virtual network rule to the data lake storage instance

# Add virtual network rule to data lake instance

$subnetInstance = Get-AzVirtualNetworkSubnetConfig -Name $subNetName -VirtualNetwork $vnetInstance

Add-AzStorageAccountNetworkRule -ResourceGroupName $resourceGroupName -Name $dataLakeAccountName -VirtualNetworkResourceId $subnetInstance.Id

1. Confirm data warehouse access to the data lake. SQL Data Warehouse is still able to access the data lake instance because it authenticates using a managed service identity.

# Issue data warehouse query to test access to data lake storage instance. This query succeeds because the data warehouse uses managed service identity authentication

Invoke-Sqlcmd -Query "SELECT \* FROM staging.dataLakeAccessTest" -ServerInstance $fullyQualifiedServerName -Database $dataWarehouseName -Username $dbCredentials.UserName -Password $dbCredentials.GetNetworkCredential().Password

**Add SQL Data Warehouse instance to virtual network**

In this section you will configure your data warehouse to allow access only from a specific virtual network. Like the data lake instance, this will be done by configuring a service endpoint within the virtual network.   
  
In your existing PowerShell session

1. Enable service endpoint for Azure SQL Data Warehouse on an existing virtual network and subnet

# Configure virtual network goal state

$subnetConfig = Set-AzVirtualNetworkSubnetConfig -Name $subNetName -VirtualNetwork $vnetInstance -AddressPrefix "10.0.0.0/24" -ServiceEndpoint "Microsoft.Sql"

# Enable service endpoint for Azure SQL Server on existing virtual network subnet

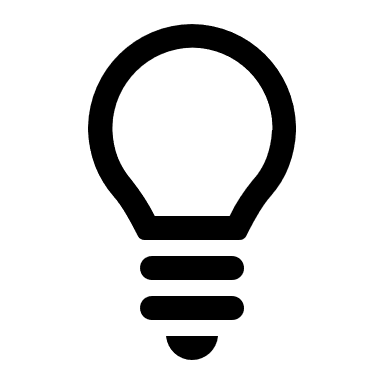
Set-AzVirtualNetwork -VirtualNetwork $subnetConfig

1. Add virtual network rule to the SQL DW instance

# Add virtual network rule to SQL Server instance

$subnetInstance = Get-AzVirtualNetworkSubnetConfig -Name $subNetName -VirtualNetwork $vnetInstance

New-AzSqlServerVirtualNetworkRule -ResourceGroupName $resourceGroupName -ServerName $serverName -VirtualNetworkRuleName $subNetName -VirtualNetworkSubnetId $subnetInstance.Id

**Feedback**

Congratulations! You have completed the lab on restricting access to data in Azure.   
Was the module helpful? Any improvements you’d suggest?   
[Take this short 1 min survey to share your thoughts with us](https://forms.office.com/Pages/ResponsePage.aspx?id=v4j5cvGGr0GRqy180BHbR90qBpvyfe5Mkaxhz8QBg_hUNjlIWU5ZR1RDTFJSTjdSOUdUS0dSNjlCNCQlQCN0PWcu).