RBAC

Last updated by | Vitor Tomaz | Aug 5, 2020 at 12:42 PM PDT

https://docs.microsoft.com/en-us/azure/dms/pre-reqs

```
$readerActions = `
"Microsoft.DataMigration/services/*/read", `
"Microsoft.Network/networkInterfaces/ipConfigurations/read"
$writerActions = `
"Microsoft.DataMigration/services/*/write", `
"Microsoft.DataMigration/services/*/delete", `
"Microsoft.DataMigration/services/*/action"
$writerActions += $readerActions
$readerRoleName = "Azure Database Migration Reader"
$contributorRoleName = "Azure Database Migration Contributor"
# TODO: replace with actual subscription IDs and add additional Ids as needed
# $subScopes = ,"/subscriptions/00000000-0000-0000-0000-000000000000/","/subscriptions/11111111-1111-1111-1111
function New-DmsReaderRole() {
    $oldRole = Get-AzureRmRoleDefinition -Name $readerRoleName
    if ($oldRole -ne $null)
    {
        """$readerRoleName"" already exists!! Delete it first if you want to recreate!!"
        return
    }
    $aRole = [Microsoft.Azure.Commands.Resources.Models.Authorization.PSRoleDefinition]::new()
    $aRole.Name = $readerRoleName
    $aRole.Description = "Lets you perform read only actions on DMS service/project/tasks."
    $aRole.IsCustom = $true
    $aRole.Actions = $readerActions
    $aRole.NotActions = @()
    $aRole.AssignableScopes = $subScopes
    #Create the role
    New-AzureRmRoleDefinition -Role $aRole
}
function New-DmsContributorRole() {
    $oldRole = Get-AzureRmRoleDefinition -Name $contributorRoleName
    if ($oldRole -ne $null)
    {
        """$contributorRoleName"" already exists!! Delete it first if you want to recreate!!"
```

```
            return
        }
        $aRole = [Microsoft.Azure.Commands.Resources.Models.Authorization.PSRoleDefinition]::new()
        $aRole.Name = $contributorRoleName
        $aRole.Description = "Lets you perform CRUD actions on DMS service/project/tasks."
        $aRole.IsCustom = $true
        $aRole.Actions = $writerActions
        $aRole.NotActions = @()
        $aRole.AssignableScopes = $subScopes
        #Create the role
        New-AzureRmRoleDefinition -Role $aRole
    }
    function Update-DmsReaderRole() {
        $aRole = Get-AzureRmRoleDefinition -Name $readerRoleName
        if ($aRole -eq $null) { throw """$readerRoleName"" not found" }
        $aRole.Actions = $readerActions
        $aRole.NotActions = @()
        $aRole.AssignableScopes += $subScopes
        # remove duplicate subscriptions if any.
        $aRole.AssignableScopes = $aRole.AssignableScopes | select -Unique
        Set-AzureRmRoleDefinition -Role $aRole
    }
    function Update-DmsConributorRole() {
        $aRole = Get-AzureRmRoleDefinition -Name $contributorRoleName
        if ($aRole -eq $null) { throw """$contributorRoleName"" not found" }
        $aRole.Actions = $writerActions
        $aRole.NotActions = @()
        $aRole.AssignableScopes += $subScopes
        # remove duplicate subscriptions if any.
        $aRole.AssignableScopes = $aRole.AssignableScopes | select -Unique
        Set-AzureRmRoleDefinition -Role $aRole
    }
    if (!(Get-Module AzureRM.Resources))
    {
        Import-Module AzureRM.Resources
```

```
}

New-DmsReaderRole

Update-DmsReaderRole

New-DmsContributorRole

Update-DmsConributorRole
```

As of now, in order to create DMS project, we will need following permissions assigned under the DMS service:

```
$role.Actions.Add("Microsoft.DataMigration/locations/operationResults/read") $role.Actions.Add("Microsoft.Data
$role.Actions.Add("Microsoft.DataMigration/register/action")
$role.Actions.Add("Microsoft.DataMigration/services/checkStatus/action")
$role.Actions.Add("Microsoft.DataMigration/services/delete")
$role.Actions.Add("Microsoft.DataMigration/services/projects/accessArtifacts/action") $role.Actions.Add("Micro
$role.Actions.Add("Microsoft.DataMigration/services/projects/files/delete")
$role.Actions.Add("Microsoft.DataMigration/services/projects/files/read") $role.Actions.Add("Microsoft.DataMig
$role.Actions.Add("Microsoft.DataMigration/services/projects/read") $role.Actions.Add("Microsoft.DataMigration
$role.Actions.Add("Microsoft.DataMigration/services/projects/write")
$role.Actions.Add("Microsoft.DataMigration/services/read")
$role.Actions.Add("Microsoft.DataMigration/services/slots/delete")
$role.Actions.Add("Microsoft.DataMigration/services/slots/read")
$role.Actions.Add("Microsoft.DataMigration/services/slots/write")
$role.Actions.Add("Microsoft.DataMigration/services/start/action")
$role.Actions.Add("Microsoft.DataMigration/services/stop/action")
$role.Actions.Add("Microsoft.DataMigration/services/write")
$role.Actions.Add("Microsoft.DataMigration/skus/read")
$role.Actions.Add("Microsoft.DataMigration/services/*")
$role.Actions.Add("Microsoft.DataMigration/services/checkStatus/*")
```

Following permission is needed under the NIC or you can add it under the resource group level.
$role.Actions.Add("Microsoft.Network/networkInterfaces/ipConfigurations/read")

In addition, we have a UI bug which prevents user with above permission to see the proper buttons in DMS and will be deployed in a week or so.

You can use this link for now if you don't want to wait

https://ms.portal.azure.com/?Microsoft_Azure_DMS=int&Microsoft_Azure_DMS_feature=mdsync&feature.canmodifystamps=true&feature.canmodifyextensions=true&clientOptimizations=false

That said, above permission set provides more privileges than just creating project in DMS(similar to the contributor role)

Our product team is working to limit the current permissions but the hot fix may take two weeks or so.

In order to create the DMS service, we will need:

1. reader role on the subscription level.

2. All permissions we had earlier. Plus, we need:

$role.Actions.Add("Microsoft.Resources/deployments/*")

$role.Actions.Add("Microsoft.Network/*/read")

$role.Actions.Add("Microsoft.Network/virtualNetworks/subnets/*")

This needs to be added under the subscription level too if the DMS and VNET are under different resource groups.

How good have you found this content?