

Microsoft Azure: Automatic Scaling of Session Hosts in Window Virtual Desktop

Reducing Costs of Desktop Hosting on Microsoft Azure Infrastructure Services

Published: January 2019  
Microsoft Corporation

**Copyright information**

This document is provided "as-is". Information and views expressed in this document, including URL and other Internet website references, may change without notice.

Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

Microsoft, Active Directory, Hyper-V, SQL Server, Windows PowerShell, Windows, and Windows Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. All other trademarks are property of their respective owners.

© 2018 Microsoft Corporation. All rights reserved.

# Introduction

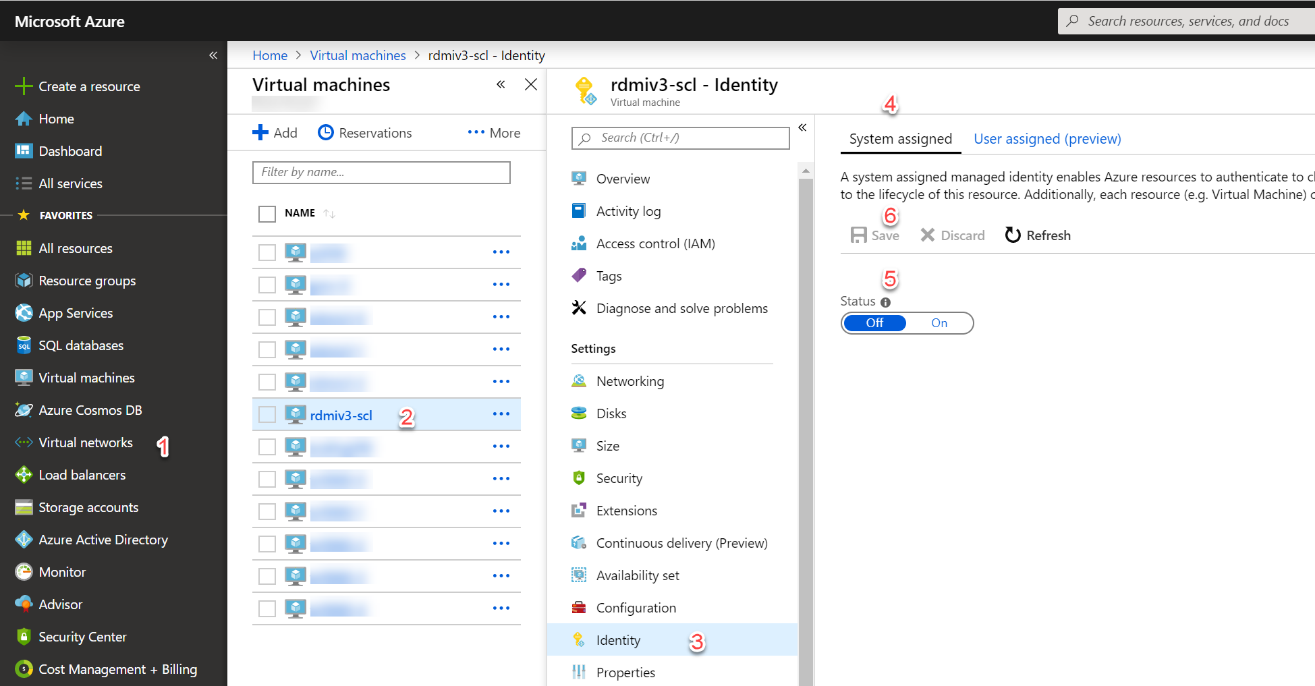
basicScale-MSI.ps1 is a sample PowerShell script that can be used as a starting point for developing a solution to automatically scale session host virtual machines.Windows Virtual Desktop deployment. Solution is distributed via [Github](https://github.com/Azure/RDS-Templates/tree/master/wvd-sh/WVD%20scaling%20script).

For many Windows Virtual Desktop deployments in Azure, virtual machine costs represent significant portion of the total Windows Virtual Desktop deployment cost. To reduce cost, the script automatically shuts down and de-allocates session host virtual machines (VMs) during off-peak usage hours and then restarts them during peak usage hours.

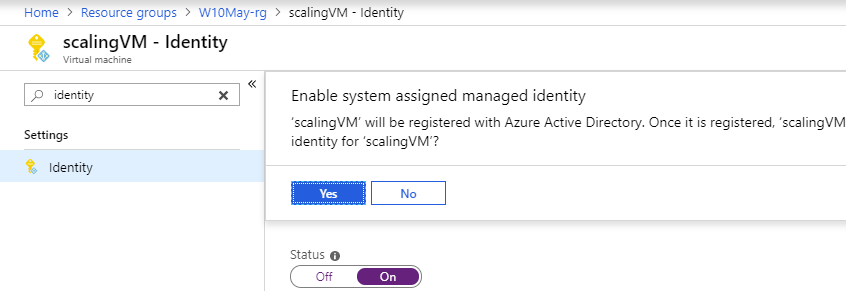
# Prerequisites

The environment to be used to execute the script must meet the following requirements:

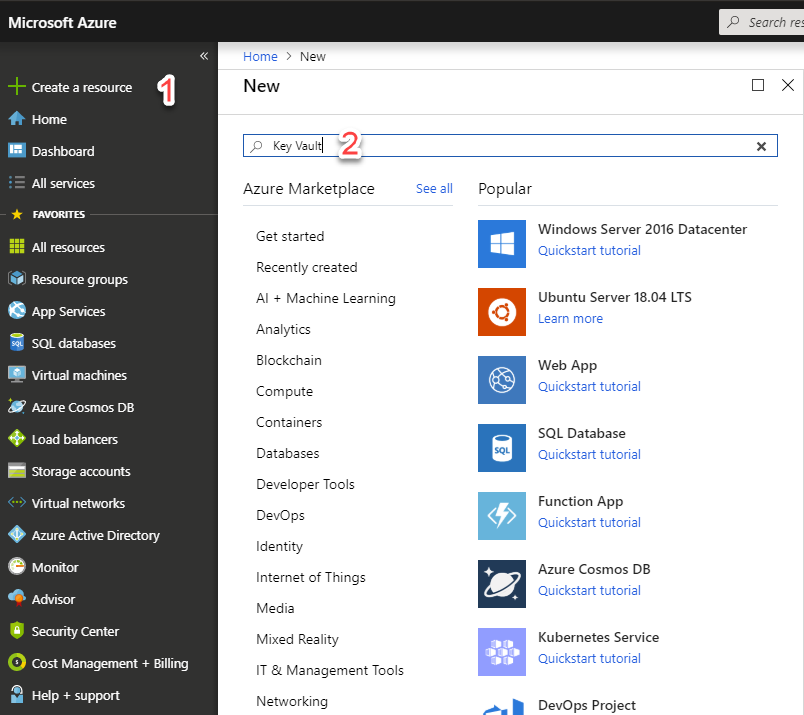
1. Windows Virtual Desktop tenant and account / [service principal](https://docs.microsoft.com/en-us/azure/virtual-desktop/create-service-principal-role-powershell) with either RDS Owner or RDS contributor.
2. Session host pool VMs configured and registered with the Windows Virtual Desktop service.
3. Additional **scaler virtual machine (VM**) that runs the scheduled task via **Task Schedule** and that has network access to session hosts.
4. Enable **managed service identity** (MSI) on the **scaler VM** running the scheduled task. To enable system-assigned managed identity on a VM that was originally provisioned without it, your account needs the [Virtual Machine Contributor](https://docs.microsoft.com/en-us/azure/role-based-access-control/built-in-roles#virtual-machine-contributor) role assignment. No additional Azure AD directory role assignments are required.
5. Sign in to the Azure portal using an account associated with the Azure subscription that contains the **Scaler VM**.
6. Navigate to the desired BM and select **Identity**.
7. Under **System assigned,** turn On **Status** and **Save** it.



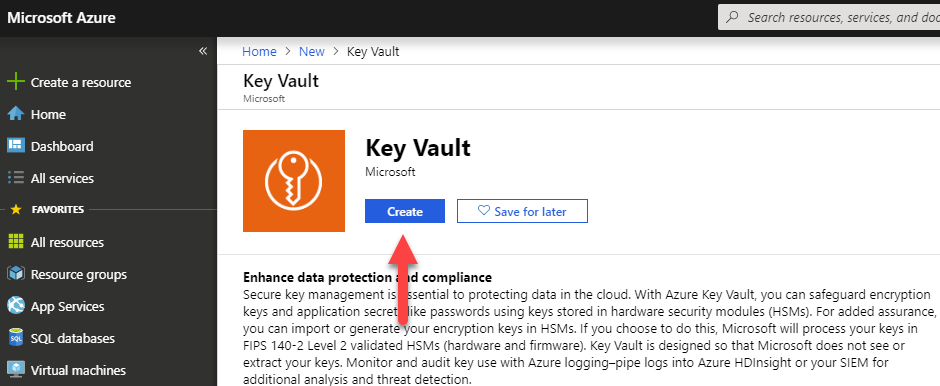
1. Confirm changes by pressing **Yes**



1. Next, create an Azure **Key Vault** and store WVD credentials in this Key vault. To set and retrieve a secret from Azure Key Vault using the azure portal follow the below sequence of steps.
2. Sign in to the Azure portal at <https://portal.azure.com>
3. Select **Create a resource** on the top-left of the Azure portal
4. Search for **Key Vault**

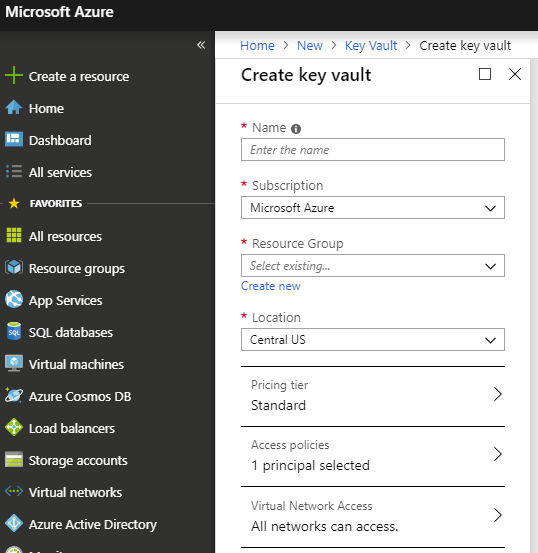


1. From the results list, choose **Key Vault**.
2. On the Key Vault section, choose **Create**.

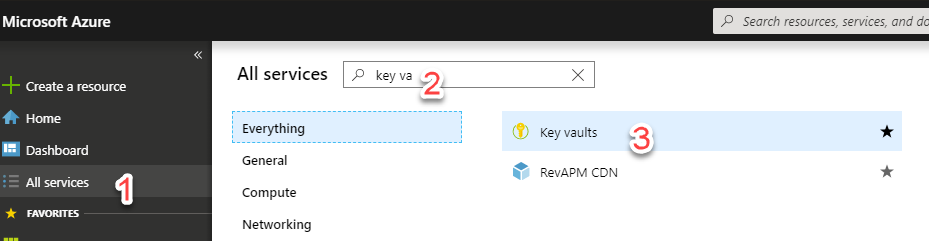


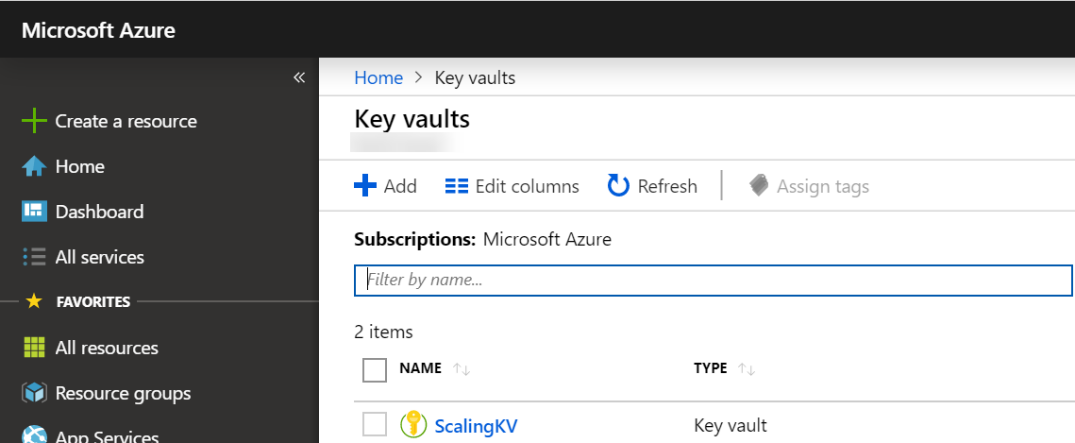
1. On the **Create key vault** section provide the following information:

* **Name**: A unique name is required. For this quick start we use Contoso-vault
* **Subscription**: Choose a subscription
* Under **Resource Group** choose Create new and enter a resource group name
* In the **Location** pull-down menu, choose a location
* Leave rest of options to their defaults and click **Create** button

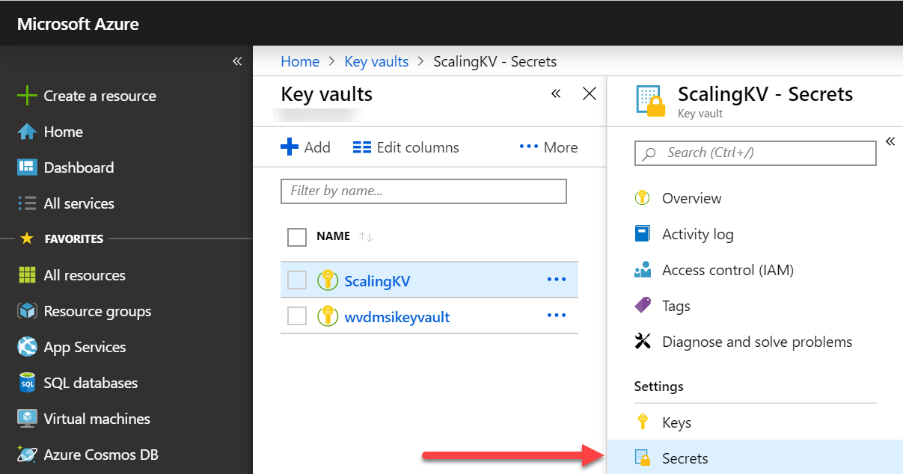


1. After deployment of the key vault has completed navigate to it.

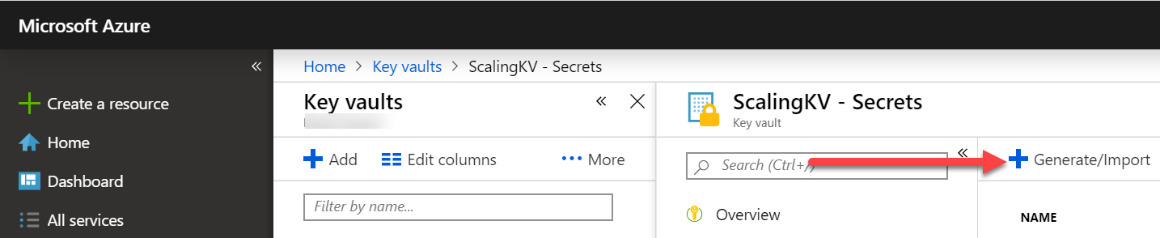




1. We are going to a password that could be used by the scaling script. The password is called WVD Tenant Admin name or WVD service Principal GUID and we store the value of Password that is WVD Tenant Admin password or WVD Service Principal Secret in it.
2. Click on desired key vault. From the **Settings** section select **Secrets.**

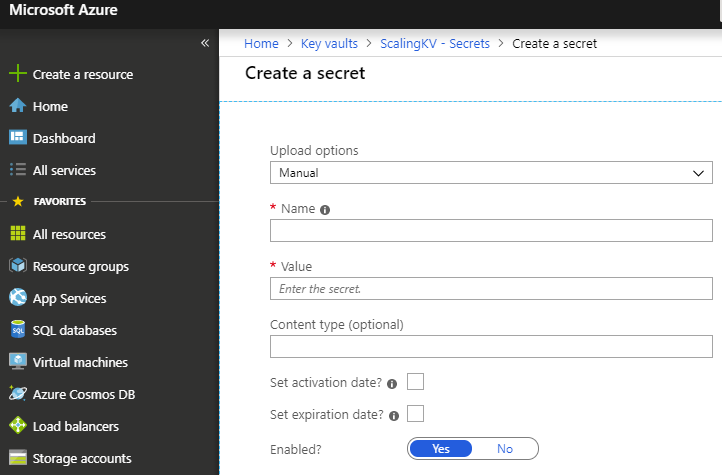


1. Click on **Generate/Import**.

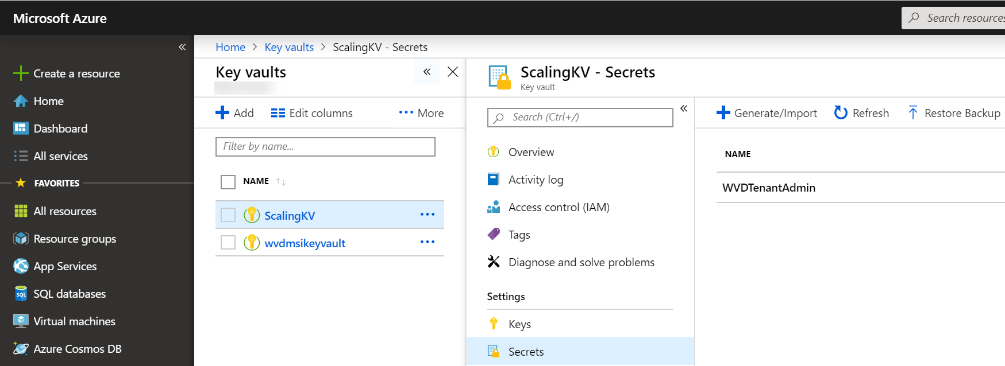


1. On the **Create a secret** screen choose the following values:

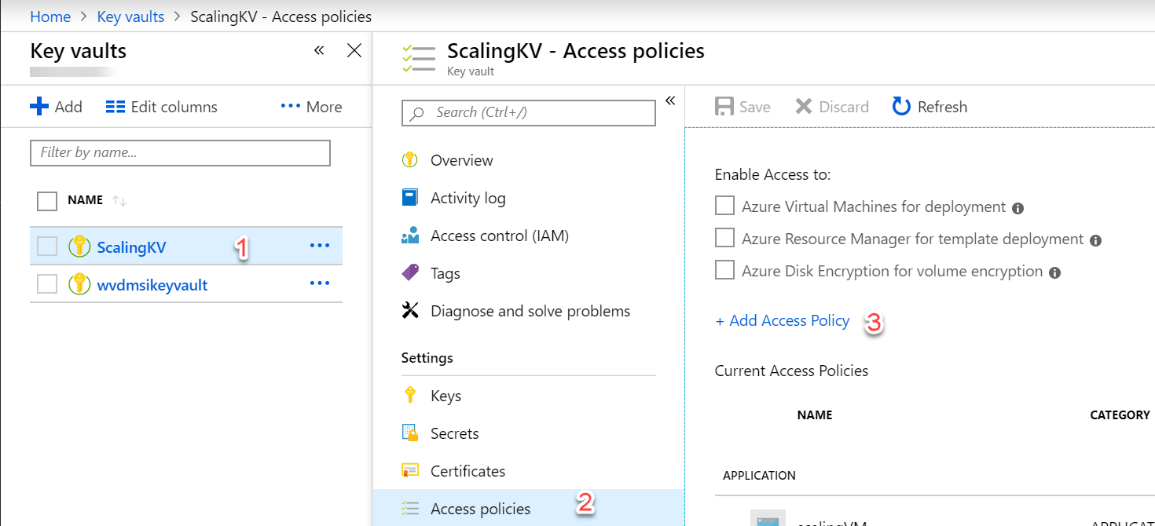
* **Upload options**: Manual
* **Name**: Provide a value (WVDTenantAdmin)
* **Value**: Provide password of WVD Tenant Admin or WVD Service Principal Secret (as configured in Azure AD)
* Leave the other values to their defaults. Click **Create**



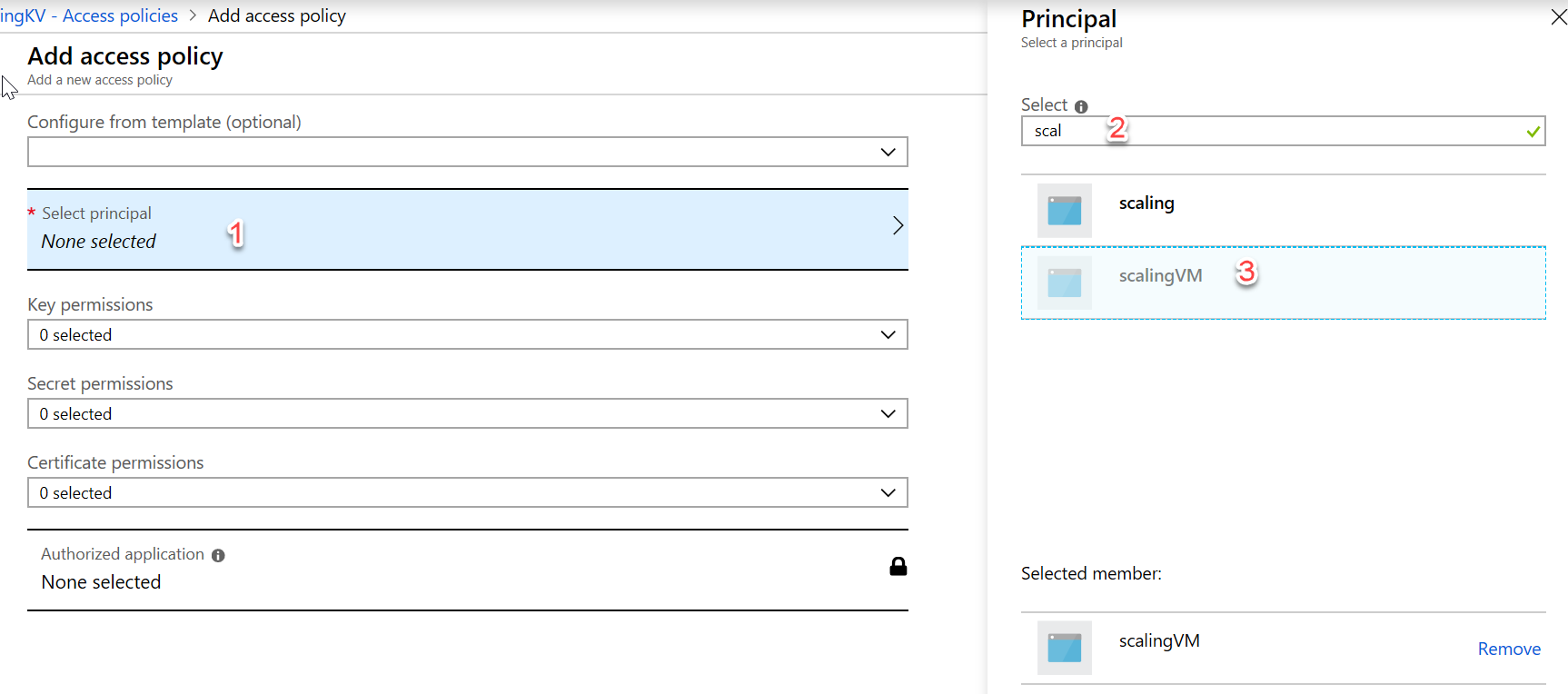
1. Once that you receive the message that the secret has been successfully created.



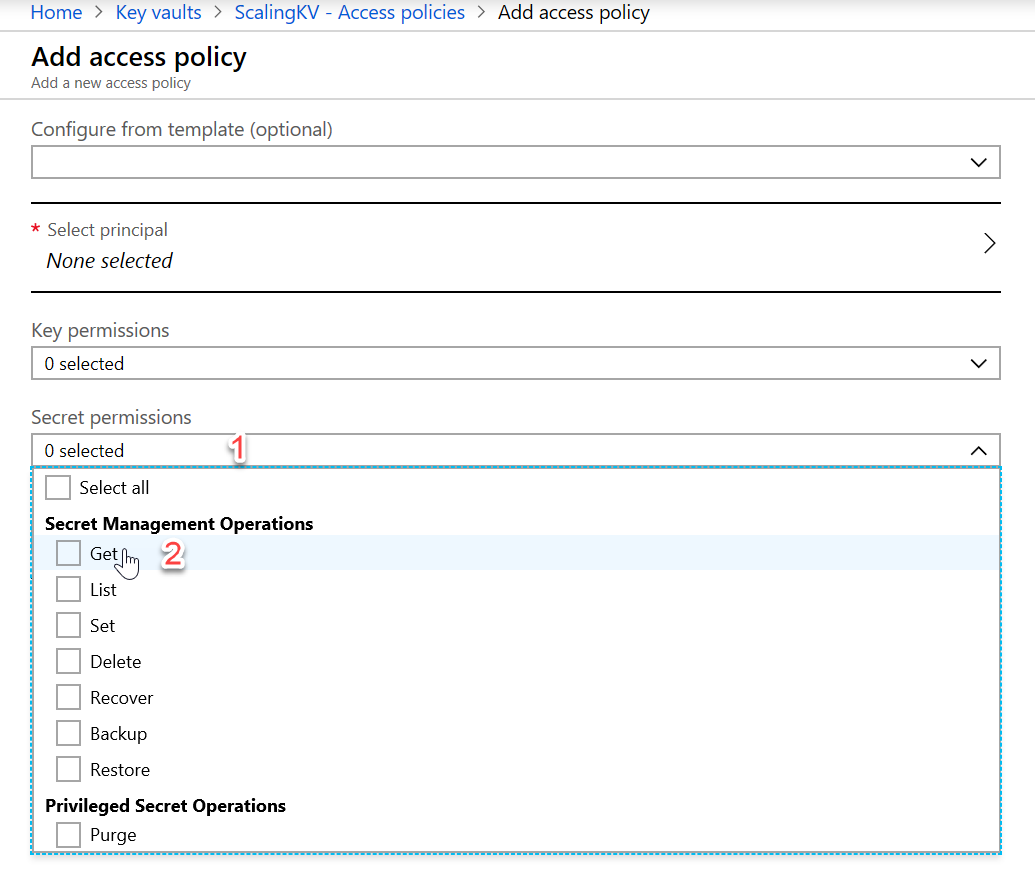
1. Once secret is created, you should provide Key Vault Access policies to service principal of **scaler VM** as below.
2. Select **Access policies** and click on **Add new** button then redirect to **Add access policy** tab.



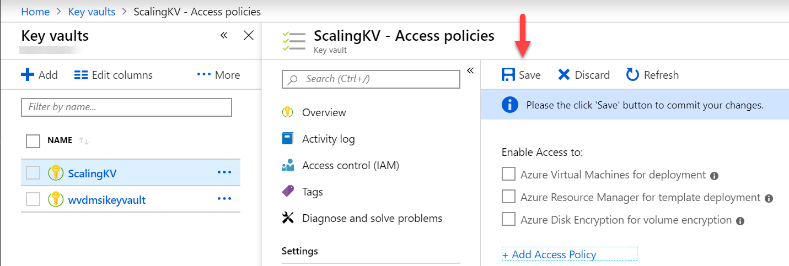
1. In **Add access policy** tab click on the **Select principal** and in the **Select** text box enter name of the **scaler VM.** Confirm selection by pressing **Select.**



1. From **Secret permission** select “Get” permission then click on **OK** button.

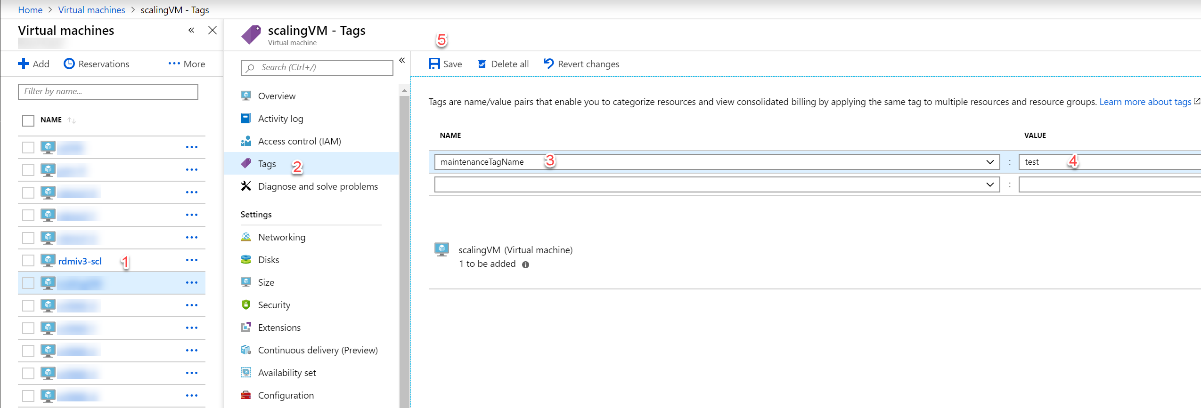


1. Click **Save** to confirm all changes.



1. If you want to keep some host servers in maintenance, then select those host servers in Azure Resource Manager Portal and add the tag with any name, use this name as value in “**maintenanceTagName**” parameter available in **Config-MSI.json** or **Config.json** file. Scaling script will check for the presence of this **“maintenanceTagName”** value with respective to the given tag name in portal, then those host servers (VMs) is not taken in consideration for any calculations or metrics.

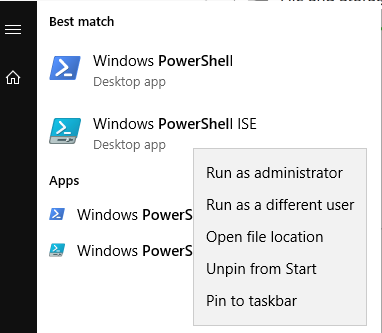
For example:



1. Select VM (host server) which you want to keep in maintenance from Azure Portal.
2. Click on **Tags**
3. Provide the **name** field for the tag, **value** field is not mandatory.
4. Click on Save button.

**Note**: The name given in (Step 3) should be used in the **Config-MSI.json** or C**onfig.json** file.

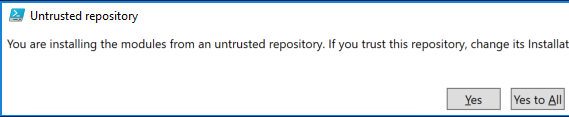
1. Install **Microsoft Azure AZ** PowerShell Module installed on the VM that is going to run the scheduled task.
2. RDP into VM that is going to run the scheduled task
3. Open **PowerShell** **ISE** as administrator



1. Execute

Install-Module Az

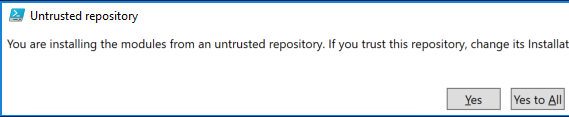
**Note**: if prompted for confirmation select **Yes to All.**



1. Windows Virtual Desktop PowerShell module installed on the VM running the scheduled task.
2. Still in **PowerShell ISE** execute

Install-Module Microsoft.RdInfra.RdPowershell

**Note**: if prompted for confirmation select **Yes to All.**



# Limitations

1. This scaling script is written to handle one host pool per scheduled task.
2. All scheduled tasks that run scaling scripts must to be configured on VM(s) that is(are) always on.
3. Accounts with MFA are not supported. It is recommended to use service principals to access the Windows Virtual Desktop service.

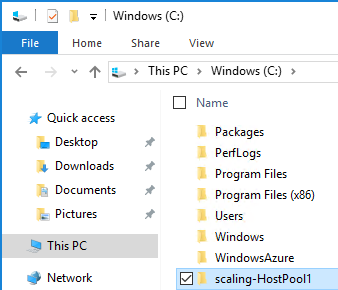
# Recommendations

1. Create a separate folder for each instance of the scaling script and its configuration.
2. Handling Maintenance and Troubleshooting when Scaling.

# Script Deployment

Use following procedure to deploy the script.

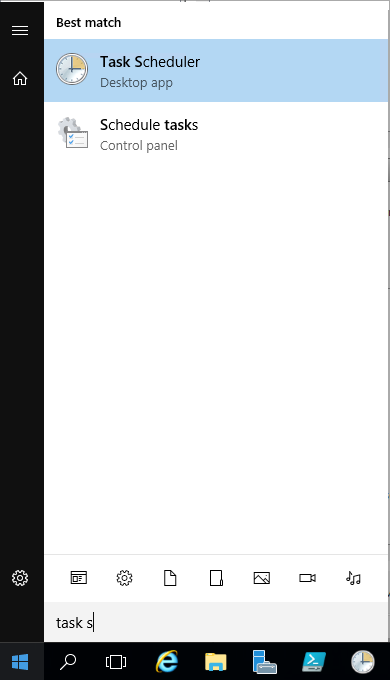
1. Logon to the VM (**scaler VM**) that is going to run the scheduled task using domain administrative account.
2. Create a folder on the scaling VM that is going to hold the scaling script and its configuration (For example, **C:\scaling-HostPool1**)



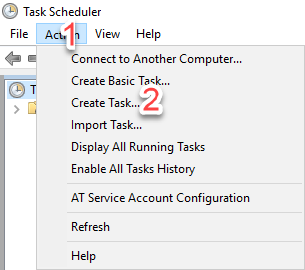
1. Download the [**basicScale-MSI.ps1**](https://github.com/Azure/RDS-Templates/blob/master/wvd-sh/WVD%20scaling%20script/basicScale-MSI.ps1)**,** [**Config-MSI.json**](https://github.com/Azure/RDS-Templates/blob/master/wvd-sh/WVD%20scaling%20script/Config-MSI.Json) files and copy them to the folder created in previous step.
2. Update the MSI Scaling Script settings in **Config-MSI.json.**

|  |  |
| --- | --- |
| **Field** | **Description** |
| AADTenantId | Azure AD Tenant ID associated with the Azure subscription where the session host VMs are running |
| currentAzureSubscriptionId | The GUID of the Azure subscription where the session host VMs are running |
| tenantGroupName | Windows Virtual Desktop tenant group |
| tenantName | Windows Virtual Desktop tenant name |
| hostPoolName | Windows Virtual Desktop host pool name |
| peakLoadBalancingType | The desired load balancing type for the host pool during peak hours. Enter either “DepthFirst” or “BreadthFirst”. |
| sessionLoadBalancingPeakHours | Session Load Balancing Peak Hours |
| maintenanceTagName | Provide the **name** field for the tag, **value** field is not mandatory. |
| keyVaultName | Key vault name where the WVD Tenant Admin Account have stored secret |
| keyVaultSecretName | The name of the secret in Azure Key Vault which contains the password for the WVD and Azure admin. |
| rdBroker | The URL to the WVD service (default value of https://rdbroker.wvd.microsoft.com) |
| username | The UPN or service principal application ID for the WVD and Azure admin. |
| isServicePrincipal | Indicates if the set of credentials being used is a service principal or a standard account. Enter “True” or “False”. |
| beginPeakTime | When peak usage time begins |
| endPeakTime | When peak usage time ends |
| timeDifferenceInHours | Time difference between local time and UTC, in hours. For example, “-7:00”. |
| sessionThresholdPerCPU | Maximum number of sessions per CPU threshold used to determine when a new session host VM needs to be started during peak hours. |
| minimumNumberOfRDSH | Minimum number of host pool VMs to keep running during off-peak usage time. |
| limitSecondsToForceLogOffUser | Number of seconds to wait before forcing users to sign out. If set to 0, users aren’t forced to sign out and VM shutdowns begin. |
| logOffMessageTitle | Title of the message sent to a user before they’re forced to sign out |
| logOffMessageBody | Body of the warning message sent to users before they’re signed out. For example, "This machine will shut down in X minutes. Please save your work and sign out.” |

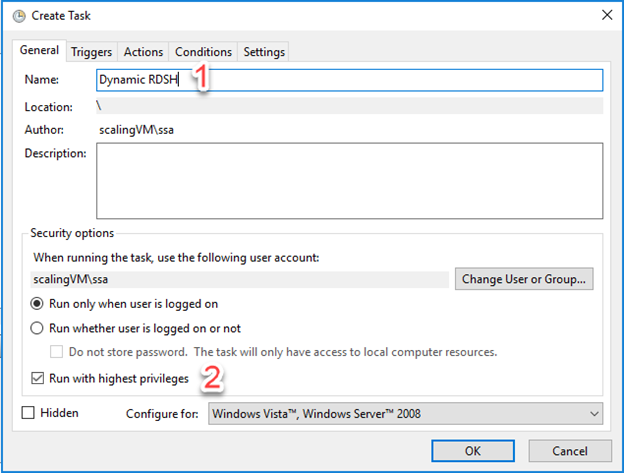
1. Configure the Task Scheduler to run the basicScaler.ps1 file at a regular interval
   1. Start **Task Scheduler**.



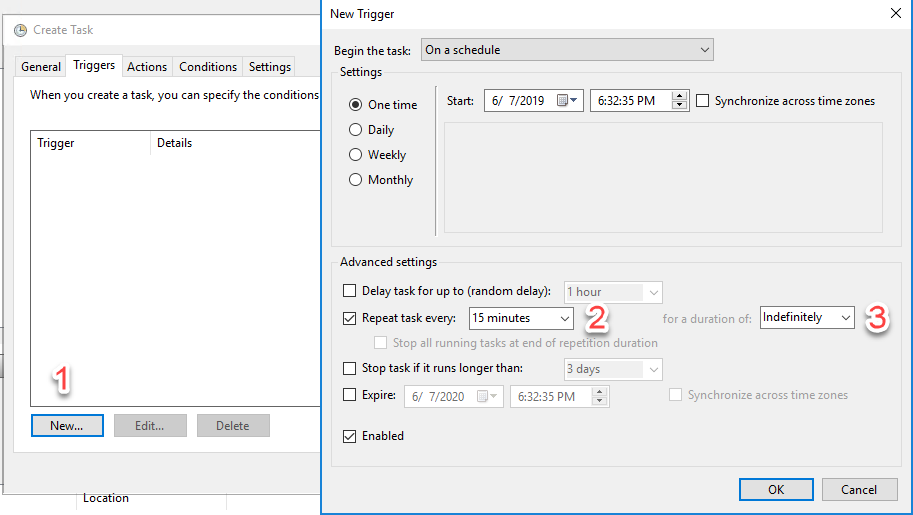
* 1. In the **Task Scheduler** window, select **Create Task …**
  2. In **the Create Task** dialog, select the **General** tab,



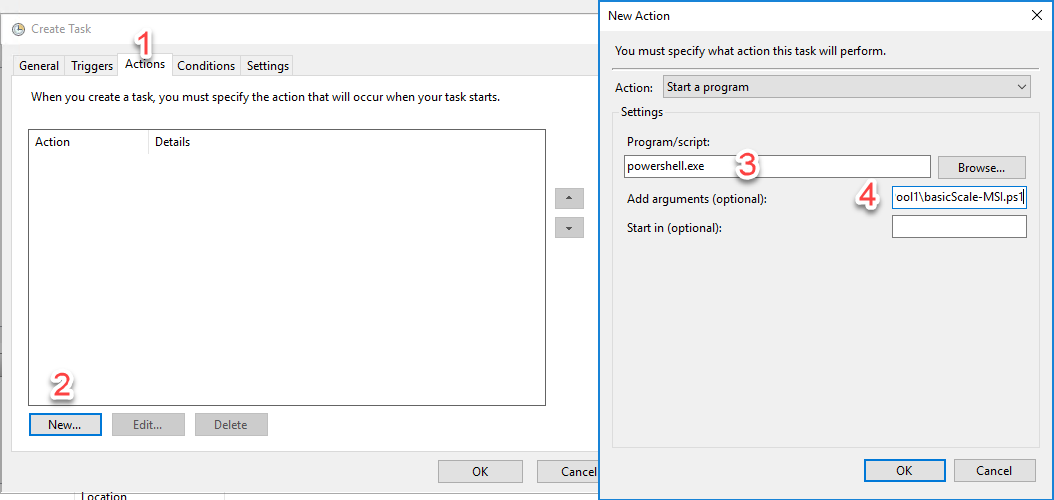
* 1. Enter a **Name: (**e.g. Dynamic RDSH), select **Run whether user is logged on or not** and **Run with highest privileges**



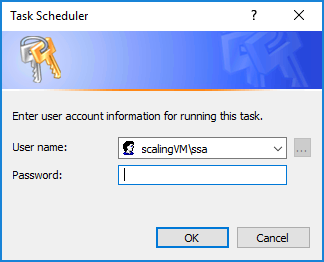
* 1. Select the **Triggers** tab and **New…**
  2. In the **New Trigger** dialog, under **Advanced settings**, check **Repeat task every** and select the appropriate period of time (e.g. **15 minutes**)
  3. The specify **Indefinitely** for **for a duration of** field



* 1. Select the **Actions** tab and **New…**
  2. In the **New Action** dialog, type **powershell.exe** in the **Program/script:** field and type **C:\scaling-HostPool1\basicScale-MSI.ps1** in the **Add arguments (optional):** field.



* 1. Select **OK** to accept defaults under the **Conditions** and **Settings** tabs
  2. Enter the password for the administrative account used to run the script



# Log Files

The script creates two log files, **WVDTenantScale.log** and **WVDTenantUsage.log**. The **WVDTenantScale.log** will log the events and errors (if any) during each execution of the script.

The **WVDTenantUsage.log** file will record the active number of cores and active number of virtual machines at each execution of the script. You can use this information to estimate the actual usage of Microsoft Azure VMs and the cost. The file is formatted as comma separated values, with each line containing the following information.

time, collection, cores, VMs

The file name can be modified to have a .csv extension, loaded into Microsoft Excel, and analyzed.

# Detailed Description

The script reads settings from a **config-MSI.json** file, including the start and end of the peak usage period during the day.

During the peak usage time, the script checks the current number of sessions and the current running RDSH capacity for each collection. It calculates if the running RDSH servers have sufficient capacity to support existing sessions based on the SessionThresholdPerCPU parameter defined in the config-MSI.json file. If not, the script starts additional RDSH servers in the collection.

During the off-peak usage time, the script determines which RDSH servers should be shutdown based on the MinimumNumberOfRDSH parameter in the config-MSI.json file. The script will set the RDSH servers to drain mode to prevent new sessions connecting to the hosts. If the LimitSecondsToForceLogOffUser parameter in the config-MSI.json file is set to a non-zero positive value, the script will notify any logged-on users to save work, wait the configured amount of time, and then force the users to logoff. Once there are no user sessions on an RDSH server, it will shut down the RDSH server.

If LimitSecondsToForceLogOffUser parameter in the config-MSI.json file is set to zero, the script will allow the session configuration setting in the collection properties to handle the logoff of user sessions. If there are any sessions on an RDSH server, it will leave the RDSH server running. If there are no sessions, the script will shut down the RDSH server.

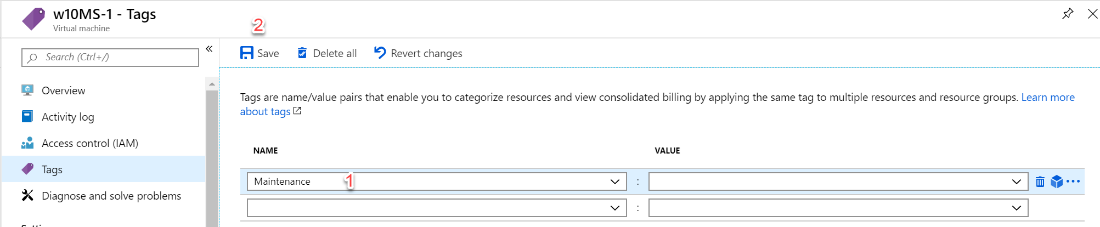
The script is designed to run periodically on the scaler VM server using Task Scheduler. You should select the appropriate time interval based on the size of your RDS environment since starting and shutting down virtual machines can take some time. We recommend running the scaling script every 15 minutes.

# Handling Maintenance and Troubleshooting when Scaling

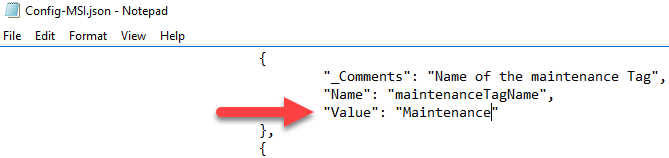
When maintenance or troubleshooting needs to be performed on a VM that is being scaled. Steps must be taken to prevent this VM from being used by the scaling logic.

## Configuration

1. Script reads the value of **maintenanceTagName** input parameter from **Config-MSI.json** file. The it will check if such tag is present on the VM.



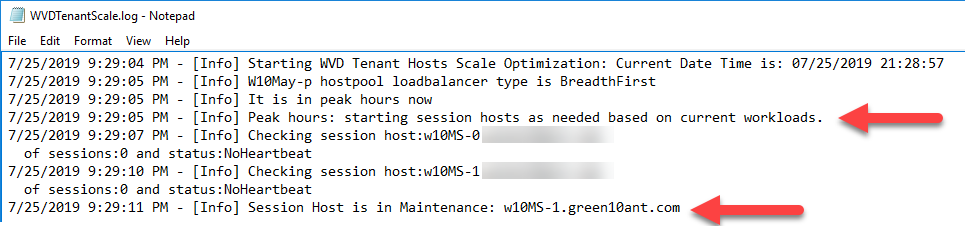
1. In **Config-MSI.json** file for input parameter **maintenanceTagName** entervalue as configured in the previous step.

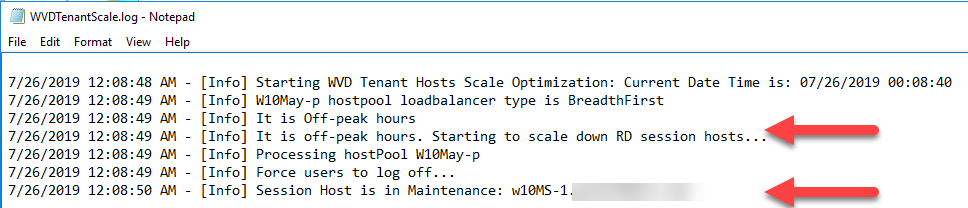


## Functionality

In peak hours script will not start the VMs which are tagged.

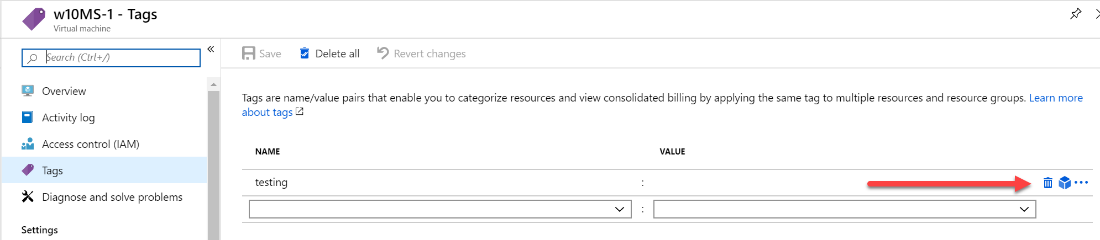
1. For your reference find the below screenshot **w10MS-1** host is skipped and not started because it is in maintenance.



1. In off peak hours script will not stop the VMs which are tagged. For your reference find the below screenshot **w10MS** VM is skipped and not stopped because it is in maintenance.

## End of Maintenance

Once maintenance is completed, admin will remove the maintenance tag in azure portal then only script will consider the VM for any calculations or metrics and start or stop them in peak hours or off-peak hours.



Version History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Description** |
| v1 | September 2014 | First version |
| v1.1 | March 2015 | Fixed comment syntax error in the xml file.  Fixed collection name comparison error. Script now works with collection names with spaces and changed collection names. |
| V2.0 | February 2017 | Updated the script to support Azure Resource Manager |
| V3.0 | July 2019 | Updated with maintenance tag functionality |