

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.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 in Windows Virtual Desktop deployment.

For many Windows Virtual Desktop deployments in Azure, the 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 with either RDS Owner or RDS contributor.
2. Session host pool VMs configured and registered with the Windows Virtual Desktop service.
3. Additional scaler VM that runs the scheduled task via Task Schedule and that has network access to session hosts.
4. Microsoft Azure PowerShell Az Module installed on the VM running the scheduled task.
5. Windows Virtual Desktop PowerShell module copied locally to the VM that is going to run the scheduled task.

# Recommendation and limitation

1. This scaling script is written to handle one host pool per instance of the scheduled task that is running the script.
2. The scheduled tasks that run scaling scripts must to be on a VM that is always on.
3. Create a separate folder for each instance of the scaling script and its configuration.
4. Accounts with MFA are not supported. It is recommended to use service principals to access the Windows Virtual Desktop service and Azure.
5. Azure's SLA guarantee apply only to VMs in an availability set. Current document describes environment with single VM that is doing the scaling, this may not meet availability requirements.

# Script Deployment

Use the following procedure to deploy the script.

1. Logon to the VM (**scaling 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**)
3. Download the **basicScaler.ps1**, **Config.xml**, and **Functions-PSStoredCredentials.ps1** files, and **PowershellModules** folderand copy them to the folder created in the previous step.
4. Create securely stored credentials
   * Open PowerShell ISE as an administrator
   * Open the edit pane and load the **Function-PSStoredCredentials.ps1**
   * Run **Set-Variable -Name KeyPath -Scope Global -Value <FolderFromStep1>**

For example, **Set-Variable -Name KeyPath -Scope Global -Value "c:\scaling-HostPool1"**

* + Run **New-StoredCredential -KeyPath $KeyPath** this will ask you to enter credentials for WVD that have permissions to query the host pool (the host pool is specified in the **config.xml).**

**Note**: if using different service principals or standard account run the above command once for each account in order to create local stored credentials.

* + Run **Get-StoredCredential -List** to confirm credentials were created successfully

1. Update the scaling script settings in **config.xml**

|  |  |
| --- | --- |
| **Field** | **Description** |
| AADTenantId | Azure AD Tenant Id that associates the subscription where the session host VMs are running. |
| currentAzureSubscriptionId | Id of the Azure subscription where the session host VMs are running |
| tenantName | Windows Virtual Desktop tenant name |
| hostPoolName | Windows Virtual Desktop host pool name |
| RDBroker | URL to the WVD service, default value <https://rdbroker.wvd.microsoft.com> |
| Username | Service principal application id (it is possible to have the same service principal as in AADApplicationId) or standard user that does not have MFA |
| isServicePrincipal | Accepted value True/False, indicates if the second set of credentials being used is a service principal or a standard account. |
| BeginPeakTime | Begin of the peak usage time |
| EndPeakTime | End of the peak usage time |
| TimeDifferenceInHours | Time difference between local time and UTC, in hours |
| SessionThresholdPerCPU | Maximum number of sessions per CPU threshold used to determine when a new RDSH server 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 logoff. If 0, don't force users to logoff |
| LogOffMessageTitle | Message title to send to a user before forcing logoff |
| LogOffMessageBody | Body of message warning users they will be logged off e.g. "Please save your work and logoff, the machine will shut down in XXX minutes”! |

1. Configure the Task Scheduler to run the basicScaler.ps1 file at a regular interval
   1. Start **Task Scheduler**.
   2. In the **Task Scheduler** window, select **Create Task …**
   3. In **the Create Task** dialog, select the **General** tab, enter a **Name: (**e.g. Dynamic RDSH), select **Run whether user is logged on or not** and **Run with highest privileges**
   4. Select the **Triggers** tab and **New…**
   5. In the **New Trigger** dialog, under **Advanced settings**, check **Repeat task every** and select the appropriate period and duration (e.g. **15 minutes** and **Indefinitely**)
   6. Select the **Actions** tab and **New…**
   7. In the **New Action** dialog, type powershell.exe in the **Program/script:** field and type C:\scaling\basicScale.ps1 in the **Add arguments (optional):** field.
   8. Select **OK** to accept defaults under the **Conditions** and **Settings** tabs
   9. 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.xml 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.xml 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.xml 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.xml 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.xml 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.

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 |