Microsoft

Desktop-as-a-Service (DaaS)
Using Windows Virtual Desktop (WVD)

Management and Operations

Prepared for:
Service Provider Partners
Oct. 2019

Prepared by:
Microsoft – **O**ne **C**ommercial **P**artner (OCP)

Contents

1. Overview

This guide illustrates the required steps for implementing Persistent and Non-persistent virtual desktops within the new Windows Virtual Desktop (WVD) Service in Microsoft Azure.

Please be advised this information is provided to help understand/summarize this process and your enterprises` implementation may contain additional customizations and/or settings that ***might not*** be covered in this document.

2. Prerequisites

Azure & Windows Active Directory Prerequisites

Before getting started, **all** items listed below **must** be checked/validated to ensure the most basic requirements are in place to proceed with executing the remaining steps in this guide.

- An Azure Active Directory
- A Windows Server Active Directory in sync with Azure Active Directory. This can be enabled through:
    - Azure AD Connect
    - Azure AD Domain Services
- An Azure subscription, containing a virtual network that either contains or is connected to the Windows Server Active Directory

General Best Practices

Since everyone's business and technical requirements vary across the board, it is always a good idea to familiarize yourselves with the standard best practices across the different Azure technologies & services.

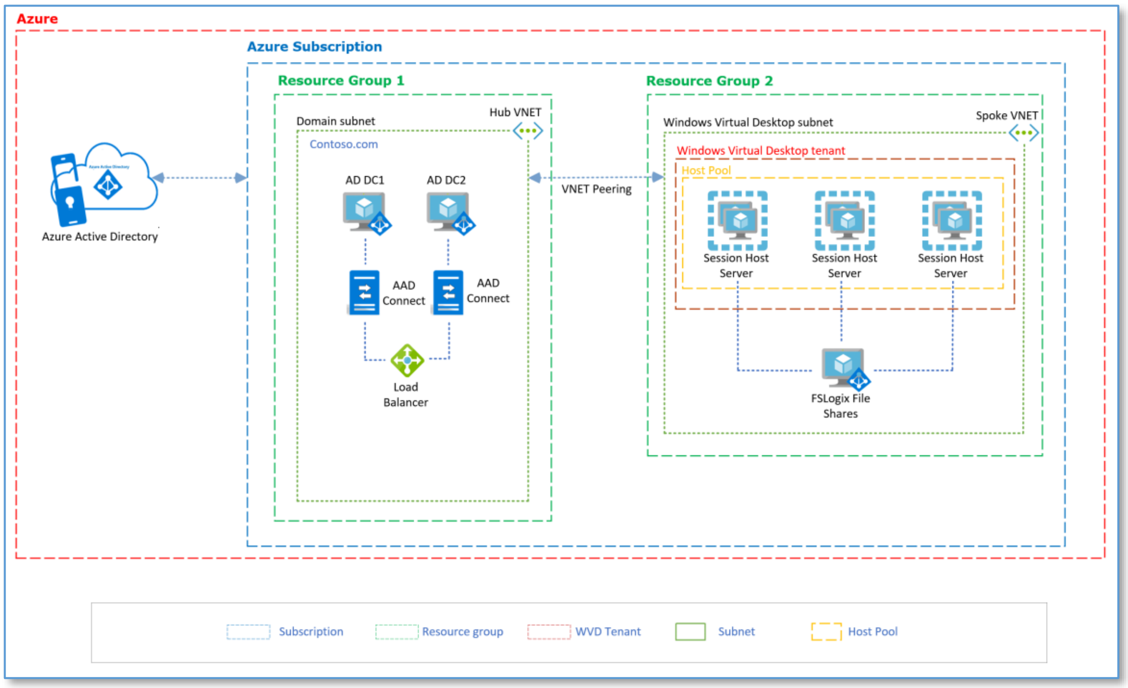
- Please follow the guidance here to maintain a consistent naming convention across your resources, unless you are already using a method.
- Azure security best practices and patterns
- Azure Active Directory Hybrid Identity best practices
- Azure identity management and access control security best practices
- Azure Networking & security Best Practices
- Azure Storage security overview
- Best practices for Azure VM security

Azure Networking

The recommendation is to design your Azure Networking using a [Hub-Spoke topology](). Consider the HUB like a DMZ deployed with your Virtual network Gateways and other security/edge appliances like Firewalls Etc. while the Spoke will act as the backend zone where your session hosts servers are deployed to and is peered with the HUB.

Azure Architectural Diagram

Below is a diagram of the Azure environment that we'll use. It shows the objects created in Azure and their relationships within the environment. In this example, the company name will be Contoso.



3. Creating a WVD Tenant

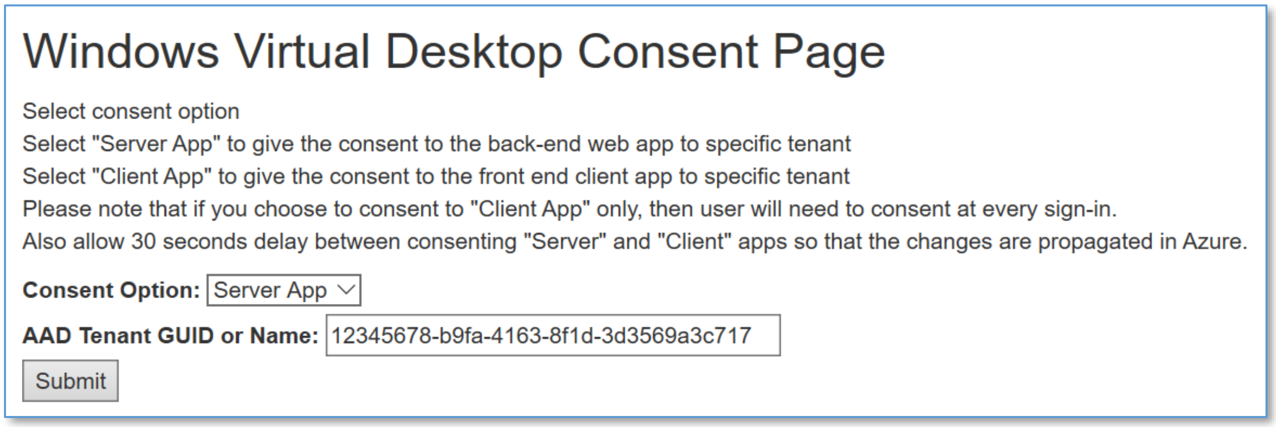
Creating a tenant in Windows Virtual Desktop Preview is the first step toward building your desktop virtualization solution. A tenant is a group of one or more host pools. Each host pool consists of multiple session hosts, running as virtual machines in Azure and registered to the Windows Virtual Desktop service.

Grant Azure Active Directory permissions to WVD

Granting permissions to the Windows Virtual Desktop service lets it query Azure Active Directory for administrative and end-user tasks. If you have already granted permissions to Windows Virtual Desktop for this Azure Active Directory instance, skip this section.

1. Open a browser and connect to the Windows Virtual Desktop consent page.
2. For **Consent Option** select **Server App**, enter the Azure Active Directory tenant name or Directory ID, and then select **Submit**.

   For Cloud Solution Provider customers, the ID is the customer's Microsoft ID from the Partner Portal. For Enterprise customers, the ID is located under **Azure Active Directory** > **Properties** > **Directory ID**.



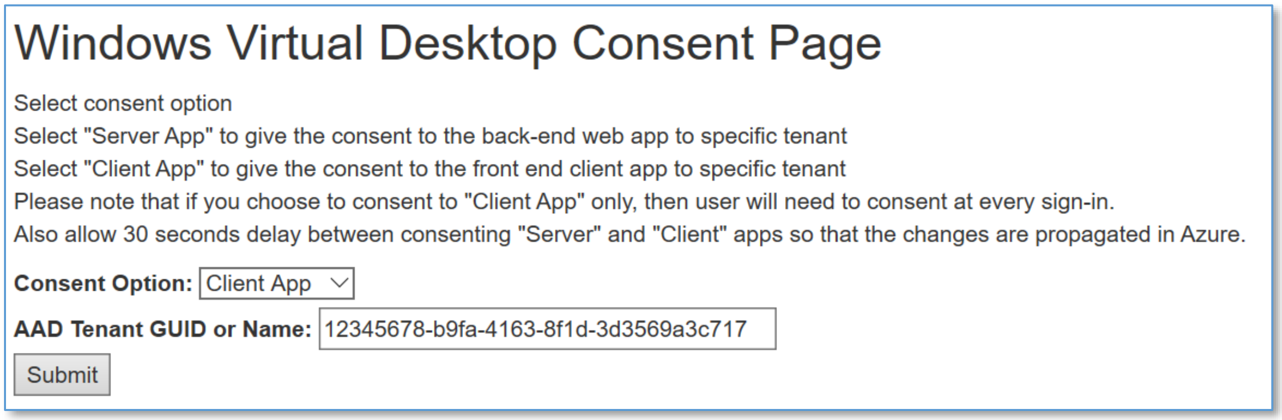
Windows Virtual Desktop Consent Page

Select consent option
Select "Server App" to give the consent to the back-end web app to specific tenant
Select "Client App" to give the consent to the front end client app to specific tenant
Please note that if you choose to consent to "Client App" only, then user will need to consent at every sign-in.
Also allow 30 seconds delay between consenting "Server" and "Client" apps so that the changes are propagated in Azure.

Consent Option: Server App
AAD Tenant GUID or Name: 12345678-b9fa-4163-8f1d-3d3569a3c717

3. Sign in to the Windows Virtual Desktop consent page with a global administrator account. For example, if you were with the Contoso organization, your account might be admin@contoso.com or admin@contoso.onmicrosoft.com.
4. Select **Accept**.
5. Wait for one minute.
6. Go back to the Windows Virtual Desktop consent page.

7. Go to **Consent Option** & select **Client App**, enter the same Azure Active Directory tenant name or Directory ID, and then select **Submit**.



Windows Virtual Desktop Consent Page

Select consent option
Select "Server App" to give the consent to the back-end web app to specific tenant
Select "Client App" to give the consent to the front end client app to specific tenant
Please note that if you choose to consent to "Client App" only, then user will need to consent at every sign-in.
Also allow 30 seconds delay between consenting "Server" and "Client" apps so that the changes are propagated in Azure.

Consent Option: Client App

AAD Tenant GUID or Name: 12345678-b9fa-4163-8f1d-3d3569a3c717

8. Sign in to the Windows Virtual Desktop consent page as global administrator, as you did in step 3.

9. Select **Accept**.

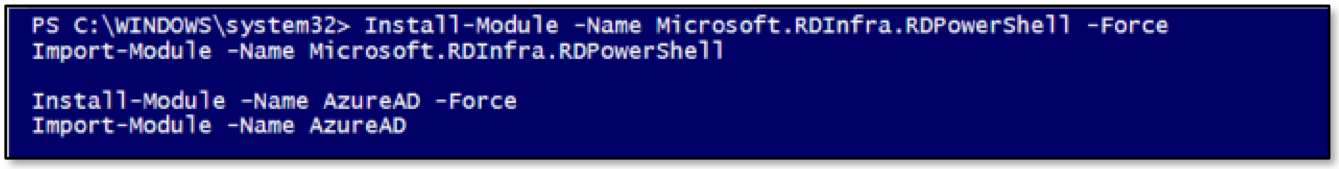
Assign a Tenant Creator user

Assigning an Azure Active Directory user the TenantCreator application role allows that user to create a Windows Virtual Desktop tenant associated with the Azure Active Directory instance. You'll need to use your global administrator account to assign the TenantCreator role.

Note: You must select a user (or a group that contains a user) that's sourced from this Azure Active Directory instance. You can't choose a guest (B2B) user or a service principal.

1. Launch PowerShell as an Administrator and run the following commands. If prompted, select “Yes to all”:

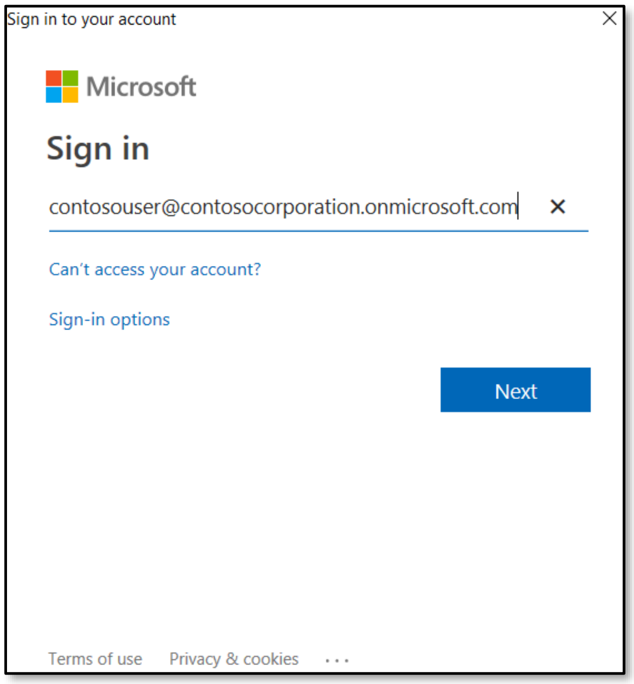
```
Install-Module -Name Microsoft.RDInfra.RDPowerShell -Force
Import-Module -Name Microsoft.RDInfra.RDPowerShell

Install-Module -Name AzureAD -Force
Import-Module -Name AzureAD
```



```
PS C:\WINDOWS\system32> Install-Module -Name Microsoft.RDInfra.RDPowerShell -Force
Import-Module -Name Microsoft.RDInfra.RDPowerShell

Install-Module -Name AzureAD -Force
Import-Module -Name AzureAD
```

2. Connect to Azure using the account you want to assign the TenantCreator role:

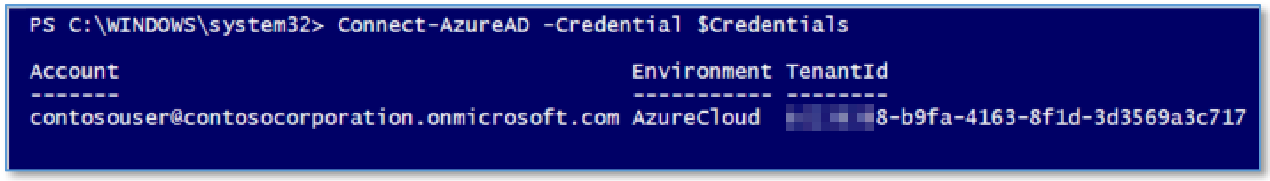
```
Connect-AzureAD -Credential $Credentials
```


Sign in to your account

Microsoft

Sign in

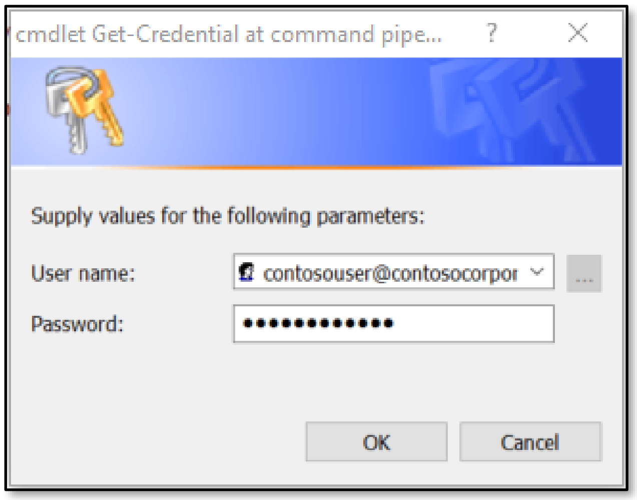
contosouser@contosocorporation.onmicrosoft.com



```
PS C:\WINDOWS\system32> Connect-AzureAD -Credential $Credentials

Account                                        Environment TenantId
-------                                        ----------- --------
contosouser@contosocorporation.onmicrosoft.com AzureCloud  [redacted]8-b9fa-4163-8f1d-3d3569a3c717
```

3. Run the below commands, using your AAD tenant ID and subscription ID, to logon to Azure: (note the ` at end lines 3 & 4)

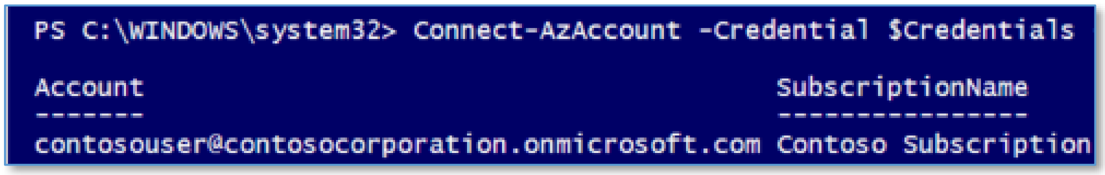
```
$Credentials = Get-Credential
$AzureAccount = Add-AzAccount -Credential $Credentials
Connect-AzAccount -Credential $Credentials `
-Tenant "12345678-b9fa-4163-8f1d-3d3569a3c717" `
-SubscriptionId "12345678-a4e8-4bab-94f7-6639ac4af7a7"
```


cmdlet Get-Credential at command pipe...

Supply values for the following parameters:

User name: contosouser@contosocorpor

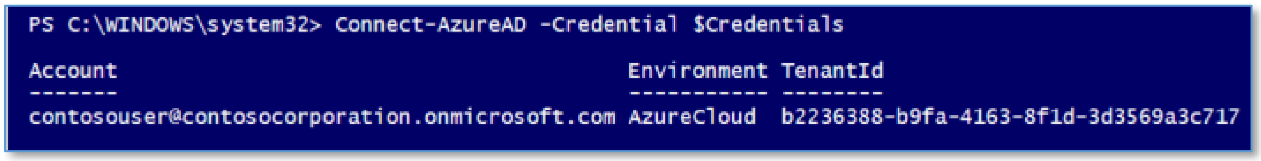
Password: •••••••••••••



```
PS C:\WINDOWS\system32> Connect-AzAccount -Credential $Credentials

Account                                          SubscriptionName
-------                                          ----------------
contosouser@contosocorporation.onmicrosoft.com Contoso Subscription
```

4. Then, connect to the Azure AD instance by running:

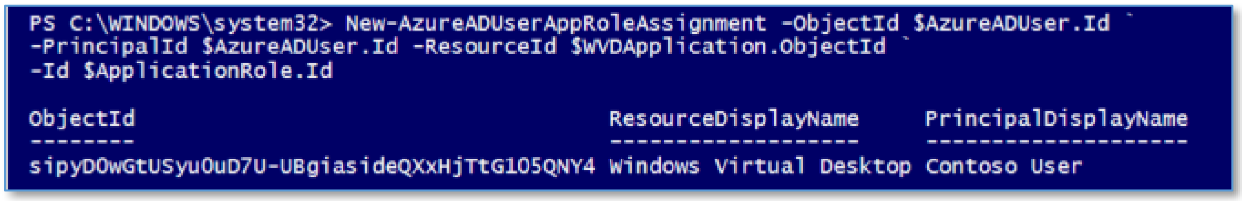
```
Connect-AzureAD -Credential $Credentials
```



```
PS C:\WINDOWS\system32> Connect-AzureAD -Credential $Credentials

Account                                        Environment TenantId
-------                                        ----------- --------
contosouser@contosocorporation.onmicrosoft.com AzureCloud  b2236388-b9fa-4163-8f1d-3d3569a3c717
```

5. Finally, execute the code below, as is, to make the assignment:

```
$WVDApplication = Get-AzureADServicePrincipal -Filter "displayName eq 'Windows
Virtual Desktop'"
$ApplicationRole = $WVDApplication.AppRoles | Where-Object {$_.DisplayName -eq
'TenantCreator'}
$AzureADUser = Get-AzADUser -UserPrincipalName $AzureAccount.Context.Account
New-AzureADUserAppRoleAssignment -ObjectId $AzureADUser.Id `
-PrincipalId $AzureADUser.Id -ResourceId $WVDApplication.ObjectId `
-Id $ApplicationRole.Id
```



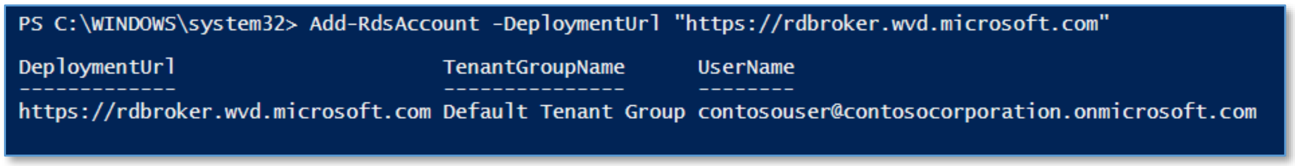
```
PS C:\WINDOWS\system32> New-AzureADUserAppRoleAssignment -ObjectId $AzureADUser.Id `
-PrincipalId $AzureADUser.Id -ResourceId $WVDApplication.ObjectId `
-Id $ApplicationRole.Id

ObjectId                                   ResourceDisplayName     PrincipalDisplayName
--------                                   -------------------     --------------------
sipyD0wGtUSyu0uD7U-UBgiasideQXxHjTtG105QNY4 Windows Virtual Desktop Contoso User
```

The user account has now been assigned the Tenant Creator role.

Create the WVD tenant

6. Run the following command to sign into Windows Virtual Desktop using the TenantCreator user account

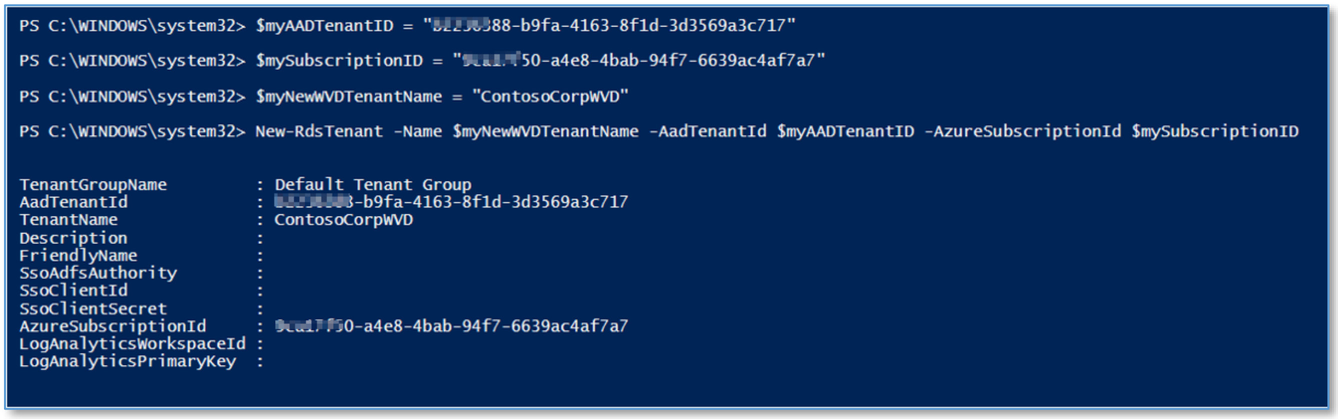
```
Add-RdsAccount -DeploymentUrl https://rdbroker.wvd.microsoft.com
```



```
PS C:\WINDOWS\system32> Add-RdsAccount -DeploymentUrl "https://rdbroker.wvd.microsoft.com"

DeploymentUrl                     TenantGroupName      UserName
-------------                     ---------------      --------
https://rdbroker.wvd.microsoft.com Default Tenant Group contosouser@contosocorporation.onmicrosoft.com
```

7. Create a Windows Virtual Desktop tenant associated with the Azure Active Directory tenant:

```
$myAADTenantID = "12345678-b9fa-4163-8f1d-3d3569a3c717"

$mySubscriptionID = "12345678-a4e8-4bab-94f7-6639ac4af7a7"

$myNewWVDTenantName = "ContosoCorpWVD"

New-RdsTenant -Name $myNewWVDTenantName -AadTenantId $myAADTenantID -
AzureSubscriptionId $mySubscriptionID
```



```
PS C:\WINDOWS\system32> $myAADTenantID = "[redacted]88-b9fa-4163-8f1d-3d3569a3c717"

PS C:\WINDOWS\system32> $mySubscriptionID = "[redacted]50-a4e8-4bab-94f7-6639ac4af7a7"

PS C:\WINDOWS\system32> $myNewWVDTenantName = "ContosoCorpWVD"

PS C:\WINDOWS\system32> New-RdsTenant -Name $myNewWVDTenantName -AadTenantId $myAADTenantID -AzureSubscriptionId $mySubscriptionID


TenantGroupName          : Default Tenant Group
AadTenantId              : [redacted]8-b9fa-4163-8f1d-3d3569a3c717
TenantName               : ContosoCorpWVD
Description              :
FriendlyName             :
SsoAdfsAuthority         :
SsoClientId              :
SsoClientSecret          :
AzureSubscriptionId      : [redacted]50-a4e8-4bab-94f7-6639ac4af7a7
LogAnalyticsWorkspaceId  :
LogAnalyticsPrimaryKey   :
```

Create & assign a WVD tenant security principal

An account may be added within Windows Virtual Desktop to facilitate automation of management and deployment tasks. The account type must be *either* AAD User or Service Principal. If the desired user account has MFA enabled, then it is required to use a Service Principal in Azure Active Directory instead, as MFA is not currently supported.

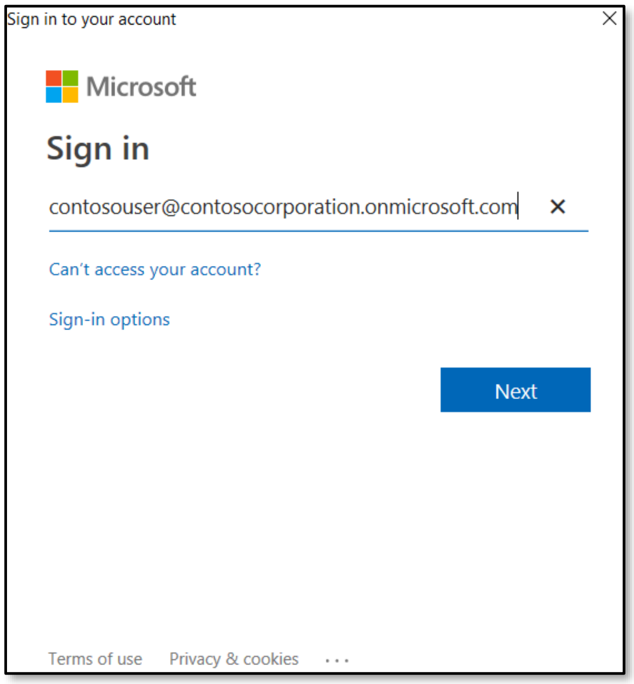
In our example, we're creating a Service Principal for that purpose. All these commands must be run within the same PowerShell session:

8. Modify the parameters below to use your Azure Active Directory Tenant ID and name:

```
$myAADTenantID = "12345678-b9fa-4163-8f1d-3d3569a3c717"
$myNewWVDTenantName = "ContosoCorpWVD" # Name of new WVD tenant
```

9. Logon to Azure as a subscription Global Administrator:

```
$aadContext = Connect-AzureAD -TenantID $myAADTenantID
```


Sign in to your account

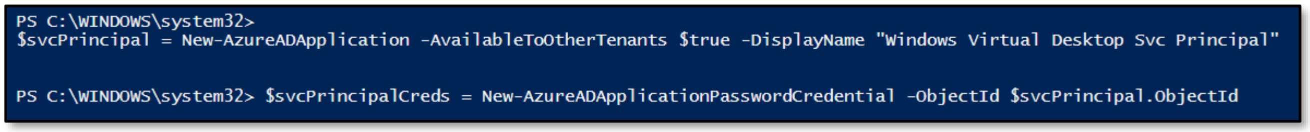
Microsoft

Sign in

contosouser@contosocorporation.onmicrosoft.com

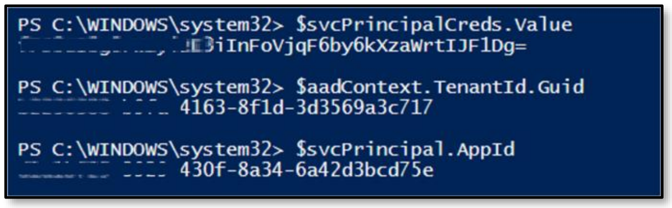
10. Create the service principal:

```
$svcPrincipal = New-AzureADApplication -AvailableToOtherTenants $true -
DisplayName "Windows Virtual Desktop Svc Principal"

$svcPrincipalCreds = New-AzureADApplicationPasswordCredential -ObjectId
$svcPrincipal.ObjectId
```


```
PS C:\WINDOWS\system32>
$svcPrincipal = New-AzureADApplication -AvailableToOtherTenants $true -DisplayName "Windows Virtual Desktop Svc Principal"

PS C:\WINDOWS\system32> $svcPrincipalCreds = New-AzureADApplicationPasswordCredential -ObjectId $svcPrincipal.ObjectId
```

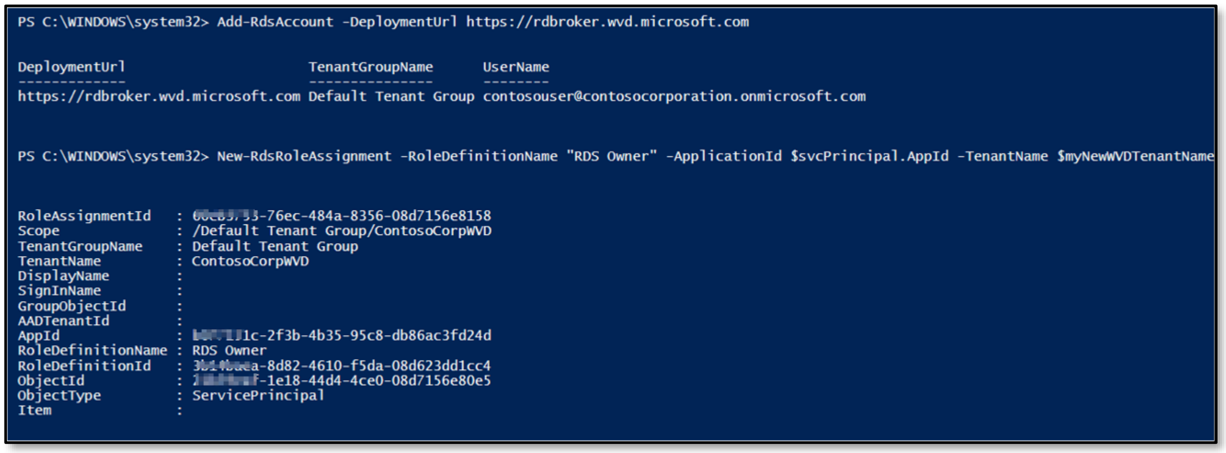
11. Run the below PS commands to display the service principals ID and password. Store them somewhere safe. Once the PowerShell session has ended, they cannot be obtained:

```
#Password
$svcPrincipalCreds.Value

#Tennant id
$aadContext.TenantId.Guid

#Application id
$svcPrincipal.AppId
```


```
PS C:\WINDOWS\system32> $svcPrincipalCreds.Value
[redacted]3iInFoVjqF6by6kXzaWrtIJF1Dg=

PS C:\WINDOWS\system32> $aadContext.TenantId.Guid
[redacted] 4163-8f1d-3d3569a3c717

PS C:\WINDOWS\system32> $svcPrincipal.AppId
[redacted] 430f-8a34-6a42d3bcd75e
```

12. Assign the RDS Owner role to the service principal:

```
Add-RdsAccount -DeploymentUrl https://rdbroker.wvd.microsoft.com

New-RdsRoleAssignment -RoleDefinitionName "RDS Owner" -ApplicationId
$svcPrincipal.AppId -TenantName $myNewWVDTenantName
```



```
PS C:\WINDOWS\system32> Add-RdsAccount -DeploymentUrl https://rdbroker.wvd.microsoft.com


DeploymentUrl                     TenantGroupName      UserName
-------------                     ---------------      --------
https://rdbroker.wvd.microsoft.com Default Tenant Group contosouser@contosocorporation.onmicrosoft.com


PS C:\WINDOWS\system32> New-RdsRoleAssignment -RoleDefinitionName "RDS Owner" -ApplicationId $svcPrincipal.AppId -TenantName $myNewWVDTenantName


RoleAssignmentId   : [redacted]3-76ec-484a-8356-08d7156e8158
Scope              : /Default Tenant Group/ContosoCorpWVD
TenantGroupName    : Default Tenant Group
TenantName         : ContosoCorpWVD
DisplayName        :
SignInName         :
GroupObjectId      :
AADTenantId        :
AppId              : [redacted]1c-2f3b-4b35-95c8-db86ac3fd24d
RoleDefinitionName : RDS Owner
RoleDefinitionId   : [redacted]a-8d82-4610-f5da-08d623dd1cc4
ObjectId           : [redacted]f-1e18-44d4-4ce0-08d7156e80e5
ObjectType         : ServicePrincipal
Item               :
```

4. Deploying Windows Virtual Desktops

Follow the steps in this section to create a host pool within the Windows Virtual Desktop tenant using an O365-optimized, multi-session VM. This includes creating a host pool in Windows Virtual Desktop, creating a resource group with VMs in an Azure subscription, joining those VMs to the Active Directory domain, and registering the VMs with Windows Virtual Desktop.

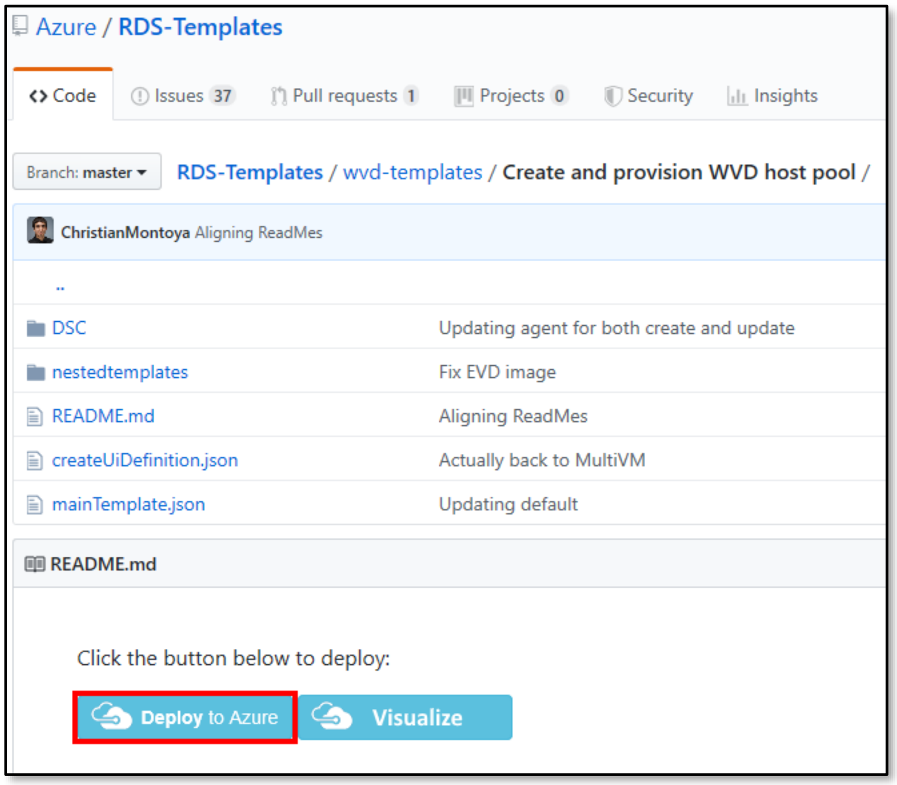
Create a Windows 10 Enterprise multi-session host pool using an ARM template

Host pools are a collection of one or more identical virtual machines within a Windows Virtual Desktop tenant environment. Each host pool can contain an app group that users can interact with as they would on a physical desktop.

You may also create a host pool via the portal using these instructions

Run the WVD Host Pool Provisioning PowerShell ARM template:

4.1 Browse to the GitHub repository here and select **Deploy to Azure**


Azure / RDS-Templates

Code | Issues 37 | Pull requests 1 | Projects 0 | Security | Insights

Branch: master RDS-Templates / wvd-templates / Create and provision WVD host pool /

ChristianMontoya Aligning ReadMes

..	
DSC	Updating agent for both create and update
nestedtemplates	Fix EVD image
README.md	Aligning ReadMes
createUiDefinition.json	Actually back to MultiVM
mainTemplate.json	Updating default

README.md

Click the button below to deploy:

4.2 Update the following parameters ONLY, leaving all others at default settings:

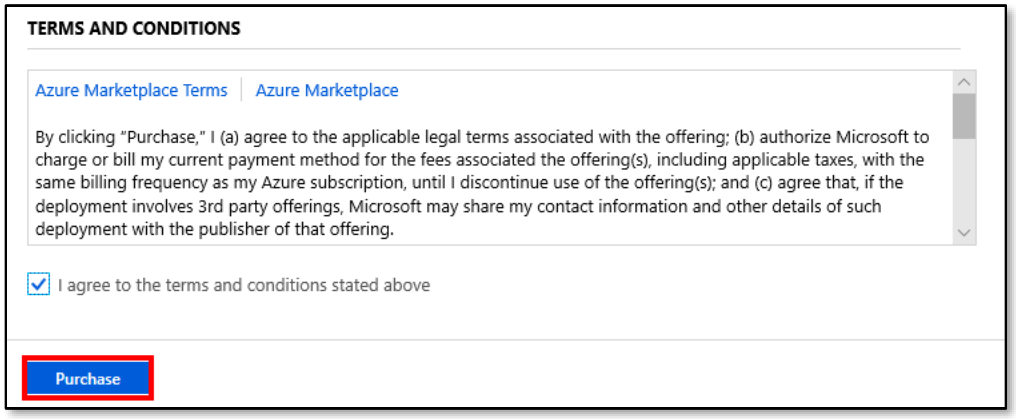
13. **Resource group** = "RG-ContosoWVD"
14. **Rdsh Number Of Instances** = 2
15. **Domain To Join** = "Contoso.local"
16. **Existing Domain UPN** = "adAdmin@contoso.local" (This UPN must have appropriate permissions to join the virtual machines to the domain and organizational unit)
17. **Existing Domain Password** = "CoNtOsOpW" (password to the account above)
18. **Existing Vnet Name** = "adVNET"
19. **Existing Subnet Name** = "adSUBNET"
20. **Virtual Network Resource Group Name** = "RG-ContosoWVD" (the appropriate ResourceGroup Name in which the VNET exists)
21. **Existing Tenant Name** = "ContosoWVD" (Provide the WVD Tenant name created earlier)
22. **Host Pool Name** = "HPPDW10ENT" (name based on what purpose the hostpool will serve. In this example the hostpool is a collection of Win 10 Enterprise multi-session.)
23. **Enable Persistent Desktop** = true
24. **Tenant Admin Upn Or Application Id**= "12345678-3929-430f-8a34-6a42d3bcd75e" (If you are creating a new host pool, this principal must be assigned either the RDS Owner or RDS Contributor role at the tenant scope (or higher). If you are registering these virtual machines to an existing host pool, this principal must be assigned either the RDS Owner or

RDS Contributor role at the host pool scope (or higher))

WARNING! You cannot enter a UPN that requires MFA to successfully authenticate. If you do, this template will create the virtual machines but fail to register them to a host pool.

25. **Tenant Admin Password** = "…sPubjyKQ80C7Qq…" (password for above account)
26. **Is Service Principal** = true (Default is false. Set to true if you are providing an ApplicationId for TenantAdminUpnorApplicationId AND providing the respective ServicePrincipal Key for TenantAdminPassword.) *We are in this example*.

4.3 Execute the ARM template by pressing the **Purchase** button


TERMS AND CONDITIONS

Azure Marketplace Terms | Azure Marketplace

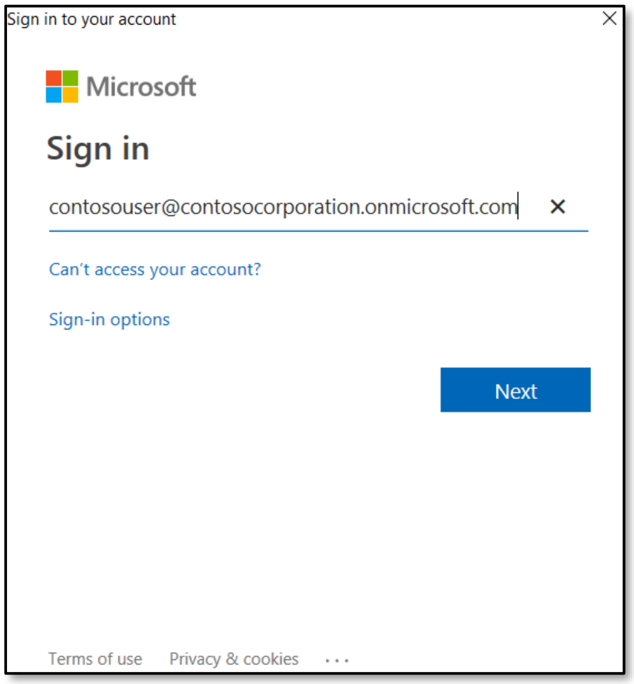
By clicking "Purchase," I (a) agree to the applicable legal terms associated with the offering; (b) authorize Microsoft to charge or bill my current payment method for the fees associated the offering(s), including applicable taxes, with the same billing frequency as my Azure subscription, until I discontinue use of the offering(s); and (c) agree that, if the deployment involves 3rd party offerings, Microsoft may share my contact information and other details of such deployment with the publisher of that offering.

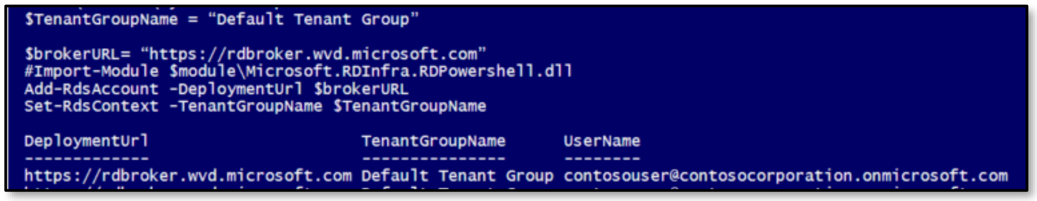
☑ I agree to the terms and conditions stated above

4.4 Once deployment completes, we will validate the newly created host pool. Open PS and connect to the WVD tenant using below commands:

```
Import-Module -Name Microsoft.RDInfra.RDPowerShell

$TenantGroupName = "Default Tenant Group"
$brokerURL= "https://rdbroker.wvd.microsoft.com"

Add-RdsAccount -DeploymentUrl $brokerURL
Set-RdsContext -TenantGroupName $TenantGroupName
```

Sign-in…





```
$TenantGroupName = "Default Tenant Group"

$brokerURL= "https://rdbroker.wvd.microsoft.com"
#Import-Module $module\Microsoft.RDInfra.RDPowershell.dll
Add-RdsAccount -DeploymentUrl $brokerURL
Set-RdsContext -TenantGroupName $TenantGroupName

DeploymentUrl                    TenantGroupName      UserName
-------------                    ---------------      --------
https://rdbroker.wvd.microsoft.com Default Tenant Group contosouser@contosocorporation.onmicrosoft.com
```

5. Choosing a Load-balancing Method

Breadth-first load balancing

Breadth-first load balancing distributes new user sessions across all available session hosts in the host pool. When configuring breadth-first load balancing, you may set a maximum session limit per session host in the host pool.

Depth-first load balancing

Depth-first load balancing distributes new user sessions to an available session host with the highest number of connections but has not reached its maximum session limit threshold. When configuring depth-first load balancing, you must set a maximum session limit per session host in the host pool.

Configure the chosen load-balancing method

Breadth-first load-balancing

Breadth-first load balancing is the default configuration for new non-persistent host pools.

- To configure a host pool to perform breadth-first load balancing *without adjusting* the maximum session limit, run the following PowerShell cmdlet:

```
$TenantName = “ContosoWVD”
$HostPoolName = “HPPDW10ENT”

 Set-RdsHostPool -TenantName $TenantName `
-HostPoolName $HostPoolName -BreadthFirstLoadBalancer
```

- To configure a host pool to perform breadth-first load balancing and to use a *new maximum session limit*, run the following PowerShell cmdlet:

```
$TenantName = “ContosoWVD”
$HostPoolName = “HPPDW10ENT”

Set-RdsHostPool -TenantName $TenantName `
-HostPoolName $HostPoolName `
-BreadthFirstLoadBalancer -MaxSessionLimit 8
```

Depth-first load balancing

To configure a host pool to perform depth-first load balancing, run the following PowerShell cmdlet:

```
$TenantName = “ContosoWVD”
$HostPoolName = “HPPDW10ENT”

Set-RdsHostPool -TenantName $TenantName `
-HostPoolName $HostPoolName `
-DepthFirstLoadBalancer -MaxSessionLimit 8
```

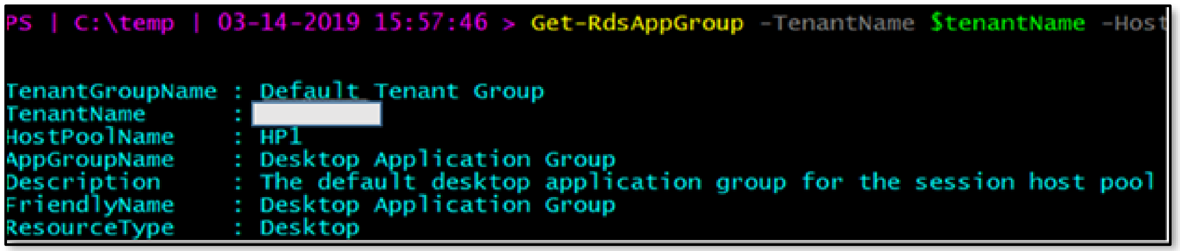
6. Managing App Groups

A default app group is automatically created for a new host pool that publishes the full desktop. In addition, you can create one or more application groups for the host pool. Host pools should be named so that it is easy to know what desktop types they contain. For example, if the host pool will host Windows 10 Multi-session VMs, then a name such as RDP-W10-MS would be a good choice.

In this section, we will create a RemoteApp AppGroup and publish individual Start menu apps.

Verify that the Default Desktop Application Group is created using below command:

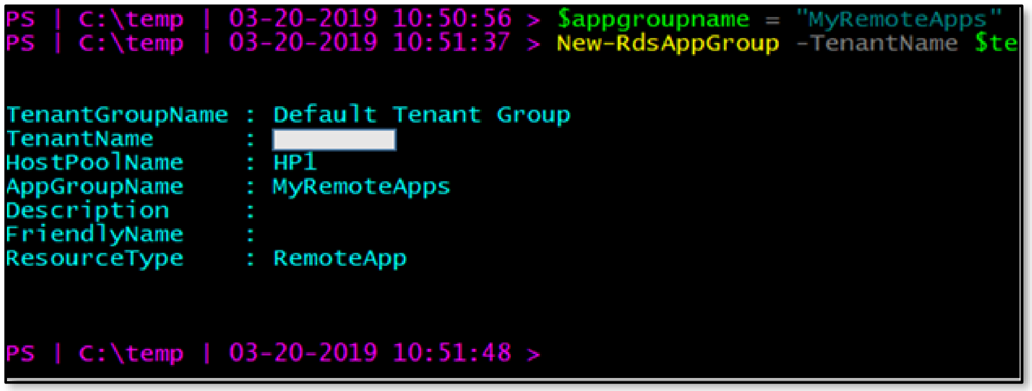
```
Get-RdsAppGroup -TenantName $TenantName -HostPoolName $HostPoolName
```



```
PS | C:\temp | 03-14-2019 15:57:46 > Get-RdsAppGroup -TenantName $tenantName -Host

TenantGroupName : Default Tenant Group
TenantName      : [redacted]
HostPoolName    : HP1
AppGroupName    : Desktop Application Group
Description     : The default desktop application group for the session host pool
FriendlyName    : Desktop Application Group
ResourceType    : Desktop
```

6.1 Now run the following PowerShell cmdlet to create a new empty RemoteApp group

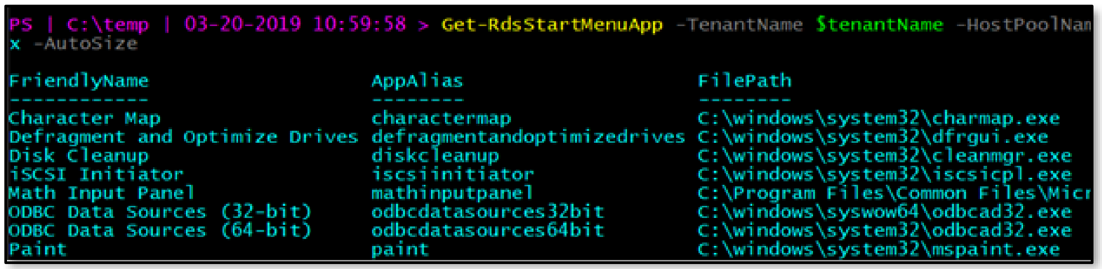
```
$appgroupname = “MyRemoteApps”
New-RdsAppGroup -TenantName $TenantName `
-HostPoolName $HostPoolName -Name $AppGroupName `
-ResourceType “RemoteApp”
```



```
PS | C:\temp | 03-20-2019 10:50:56 > $appgroupname = "MyRemoteApps"
PS | C:\temp | 03-20-2019 10:51:37 > New-RdsAppGroup -TenantName $te

TenantGroupName : Default Tenant Group
TenantName      : [redacted]
HostPoolName    : HP1
AppGroupName    : MyRemoteApps
Description     :
FriendlyName    :
ResourceType    : RemoteApp


PS | C:\temp | 03-20-2019 10:51:48 >
```

6.2 Run the following cmdlet to see a list of start menu apps available on the host pool's virtual machine image.

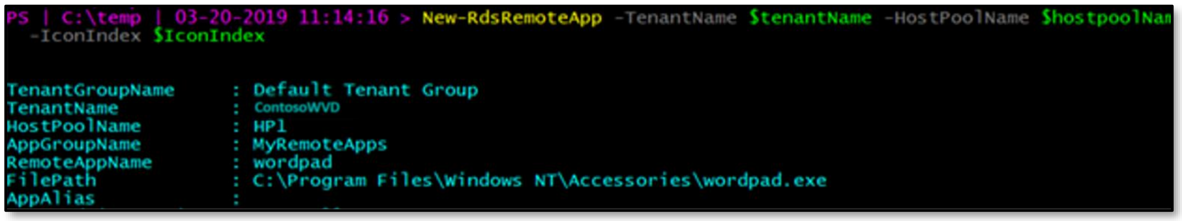
```
Get-RdsStartMenuApp -TenantName $TenantName `
-HostPoolName $HostPoolName `
-appgroupname $AppGroupName | FT `
FriendlyName,AppAlias,FilePath,IconPath,IconIndex -AutoSize
```



```
PS | C:\temp | 03-20-2019 10:59:58 > Get-RdsStartMenuApp -TenantName $tenantName -HostPoolNam
x -AutoSize

FriendlyName                 AppAlias                     FilePath
------------                 --------                     --------
Character Map                charactermap                 C:\windows\system32\charmap.exe
Defragment and Optimize Drives defragmentandoptimizedrives C:\windows\system32\dfrgui.exe
Disk Cleanup                 diskcleanup                  C:\windows\system32\cleanmgr.exe
iSCSI Initiator              iscsiinitiator               C:\windows\system32\iscsicpl.exe
Math Input Panel             mathinputpanel               C:\Program Files\Common Files\Micr
ODBC Data Sources (32-bit)   odbcdatasources32bit         C:\windows\syswow64\odbcad32.exe
ODBC Data Sources (64-bit)   odbcdatasources64bit         C:\windows\system32\odbcad32.exe
Paint                        paint                        C:\windows\system32\mspaint.exe
```

6.3 Run the following cmdlet to publish a new WordPad RemoteApp to the application group. Using the values listed from the step before, run again for each app you want to publish.

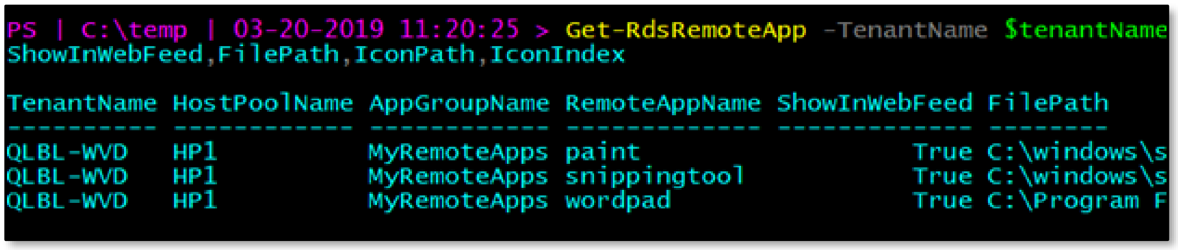
```
$AppAlias = “wordpad”
$FilePath = “C:\Program Files\Windows NT\Accessories\wordpad.exe”
$IconPath = “C:\Program Files\Windows NT\Accessories\wordpad.exe”
$IconIndex = 0

New-RdsRemoteApp -TenantName $TenantName -HostPoolName $HostPoolName `
-appgroupname $AppGroupName -Name $AppAlias -Filepath $FilePath `
-IconPath $IconPath -IconIndex $IconIndex
```



```
PS | C:\temp | 03-20-2019 11:14:16 > New-RdsRemoteApp -TenantName $tenantName -HostPoolName $hostpoolNam
  -IconIndex $IconIndex

TenantGroupName         : Default Tenant Group
TenantName              : ContosoWVD
HostPoolName            : HP1
AppGroupName            : MyRemoteApps
RemoteAppName           : wordpad
FilePath                : C:\Program Files\Windows NT\Accessories\wordpad.exe
AppAlias                :
```

6.4 To verify that the app was published, run the following cmdlet.

```
Get-RdsRemoteApp -TenantName $tenantName -HostPoolName $hostpoolName `
-AppGroupName $appgroupname | FT `
IconIndex,RemoteAppName,TenantName,HostPoolName,AppGroupName,ShowInWebFeed
```



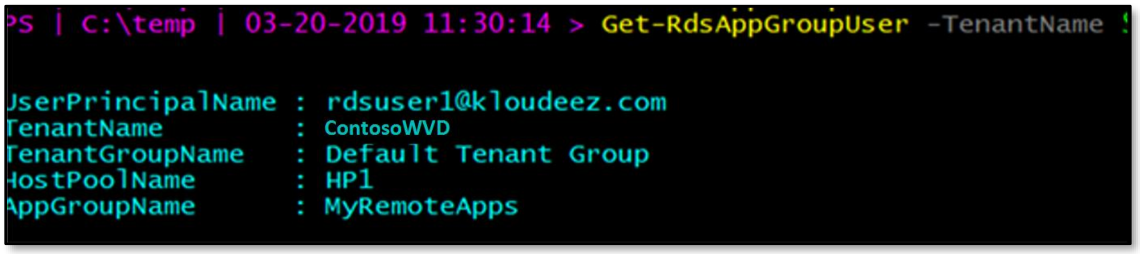
```
PS | C:\temp | 03-20-2019 11:20:25 > Get-RdsRemoteApp -TenantName $tenantName
ShowInWebFeed,FilePath,IconPath,IconIndex

TenantName HostPoolName AppGroupName RemoteAppName ShowInWebFeed FilePath
---------- ------------ ------------ ------------- ------------- --------
QLBL-WVD   HP1          MyRemoteApps paint                  True C:\windows\s
QLBL-WVD   HP1          MyRemoteApps snippingtool           True C:\windows\s
QLBL-WVD   HP1          MyRemoteApps wordpad                True C:\Program F
```

6.5 Run the following cmdlet to grant users access to the remote apps in the app group:

```
$AppGroupName = “MyRemoteApps”
$upn = “rdsuser1@contoso.com”
Add-RdsAppGroupUser -TenantName $TenantName -HostPoolName $HostPoolName `
-AppGroupName $AppGroupName -UserPrincipalName $upn
```

Verify that the ACL has been applied using:

```
Get-RdsAppGroupUser -TenantName $TenantName `
-HostPoolName $HostPoolName -AppGroupName $AppGroupName
```



```
PS | C:\temp | 03-20-2019 11:30:14 > Get-RdsAppGroupUser -TenantName $

UserPrincipalName : rdsuser1@kloudeez.com
TenantName        : ContosoWVD
TenantGroupName   : Default Tenant Group
HostPoolName      : HP1
AppGroupName      : MyRemoteApps
```

7. Automatic Scaling of WVD Session Hosts

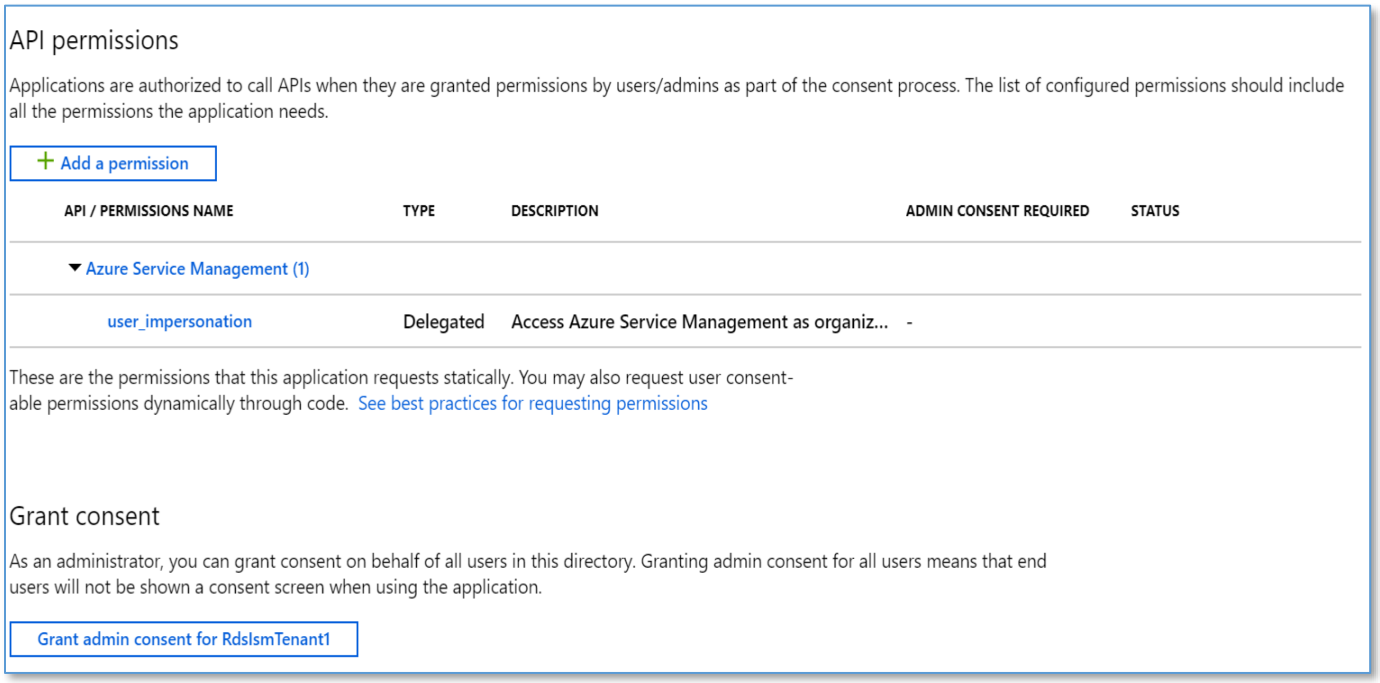
For many Windows Virtual Desktop deployments in Azure, the virtual machine costs of the Windows Virtual Desktop session host VM represent the most significant portion of the total deployment cost. To reduce cost, the script below (scaledeployment.ps1) automatically shuts down and de-allocates RDSH server VMs during off-peak usage hours and then restarts them during peak usage hours.

Prerequisites

The environment used to execute the scaledeployment.ps1 script must meet the folowing requirements.

- An Azure service principal must be enabled with appropriate permissions: Min. Role as "Contributor" or "Owner" to run the script, access Azure resources and access WVD hostpool resources. (Follow this link to create service principal.)

- Enable API permission of "Azure Service Management" to Azure service principal:



API permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs.

API / PERMISSIONS NAME	TYPE	DESCRIPTION	ADMIN CONSENT REQUIRED	STATUS
▾ Azure Service Management (1)				
user_impersonation	Delegated	Access Azure Service Management as organiz...	-	

These are the permissions that this application requests statically. You may also request user consent-able permissions dynamically through code. See best practices for requesting permissions

Grant consent

As an administrator, you can grant consent on behalf of all users in this directory. Granting admin consent for all users means that end users will not be shown a consent screen when using the application.

- PowerShell 5.0 or higher installed on your local machine.
- Microsoft Azure AzureRm PowerShell Module installed on your local machine.

Deploying the scaling script

Use the following procedure to deploy the script.

- Run the Windows PowerShell Console in Administrator mode.
- Execute the following cmdlet to download the script to local file system

```
Invoke-WebRequest -Uri "https://raw.githubusercontent.com/Azure/RDS-Templates/ptg-
wvdautoscaling-automation/wvd-templates/wvd-scaling-script/wvdscaling-
automation/scaledeployment.ps1" | Out-File “.\scaledeployment.ps1”
```

- Provide required parameter values and execute **scaledeployment**.**ps1** PowerShell Script.

```
.\scaledeployment.ps1 -AzureADApplicationId “12345678-3929-430f-8a34-6a42d3bcd75e" `
-AzureAdApplicationSecret "123456780rWzyizE3iInFoVjqF6by6kXzaWrtIJF1Dg=" `
-AADTenantID “12345678-b9fa-4163-8f1d-3d3569a3c717” `
-SubscriptionID “12345678-a4e8-4bab-94f7-6639ac4af7a7” `
-TenantGroupName “Default Tenant Group” `
-TenantName “ContosoCorpWVD” `
-HostPoolName “HPPDW10ENT” `
-PeakLoadBalancingType “” `
-MaintenanceTagName “” `
-RecurrenceInterval “15” `
-WorkspaceName “” `
-BeginPeakTime “9:00” `
-EndPeakTime “18:00” `
-TimeDifference “+5:30” `
-SessionThresholdPerCPU 6 `
-MinimumNumberOfRDSH 4 `
-LimitSecondsToForceLogOffUser 20 `
-LogOffMessageTitle “System Under Maintenance” `
-LogOffMessageBody “Please save your work and logoff!”
-Location “West US2”
```

Parameter descriptions:

Field	**Description**
*AzureADApplicationId	ApplicationId created and configured for the service principal.
*AzureADApplicationSecret	Secret configured for ApplicationId.
*AADTenantID	Azure Tenant ID that is associated with the subscription where session host VMs run.
*SubscriptionID	Azure Subscription ID where the session host VMs run. Required resources are created under the same subscription.
*TenantGroupName	Windows Virtual Desktop tenant group

*TenantName	Windows Virtual Desktop tenant name
*HostPoolName	Windows Virtual Desktop host pool name
*PeakLoadBalancingType	Hostpool session Load Balancing Type in peak hours
*MaintenanceTagName	Provide the name field for the tag, value field is not mandatory.
*RecurrenceInterval	Scheduler job will run recurrence interval basis, so provide recurrence in minutes. Repeat the job on selected period (e.g. 15 minutes).
*WorkspaceName	Name of the log analytics workspace to store basicScript.ps1 logs
*BeginPeakTime	When peak usage time begins
*EndPeakTime	When peak usage time ends
*TimeDifference	Time difference between local time and UTC, in hours
*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.
*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."
*Location	Name of the Location for to create azure automation account WVD autoscaling required resources. By default, location is "South Central US".

The Azure scheduler will create multiple instances of scheduler jobs (one per HostPool) in the job collection.

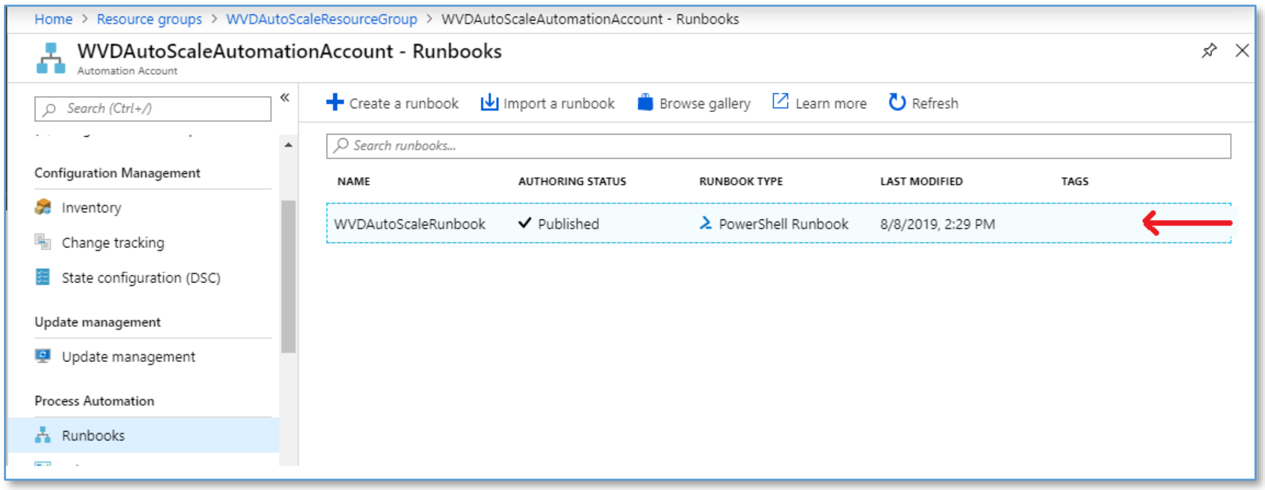
Below are additional parameters utilized for creating resources.

1. Resource Group with **default** name "**WVDAutoScaleResourceGroup**", can also be a customized name where automation account exists. Input param.
2. Automation Account with **default** name "**WVDAutoScaleAutomationAccount**" can also be a customized name. Input param.
3. Automation Account Runbook with name "**WVDAutoScaleRunbook**"
4. Automation Account Webhook with name "**WVDAutoScaleWebhook**" and generates the Webhook URI and then store it into Automation Account variable.
5. Azure Scheduler Job collection with name "**WVDAutoScaleSchedulerJobCollection**"

Azure Scheduler will create a job instance for each HostPoolName, so, ensure that each host pool has a unique name, as the host pool name is used as a prefix to the Job name (for example: HostpoolName-Job).

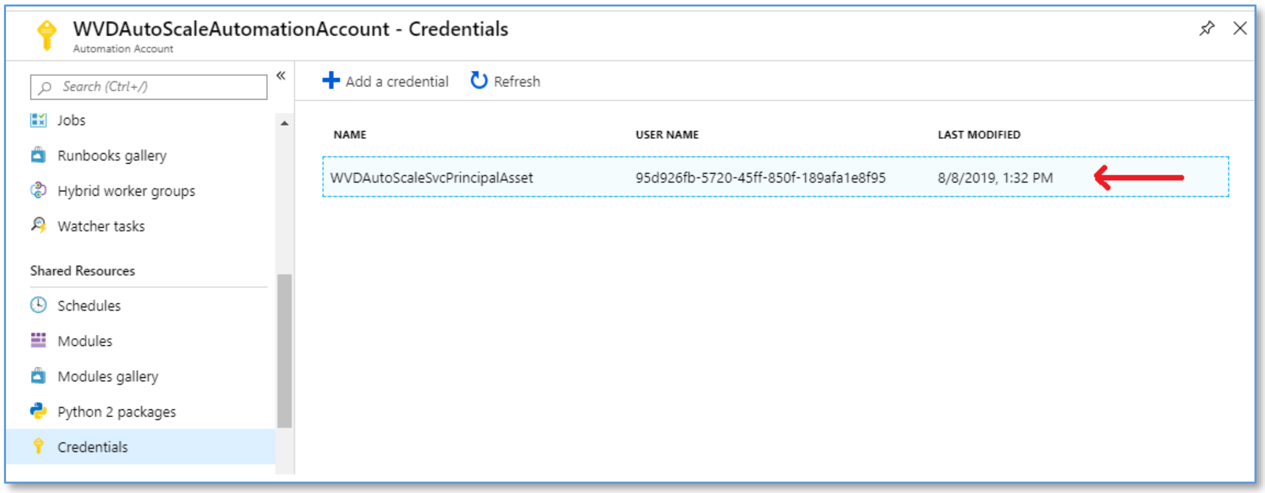
Executing the **Scaledeployment.ps1** script produces the following results:

1. Publishes the **basicScale.ps1** file into "Azure automation account Runbook" using **runbookCreationTemplate.json** template.



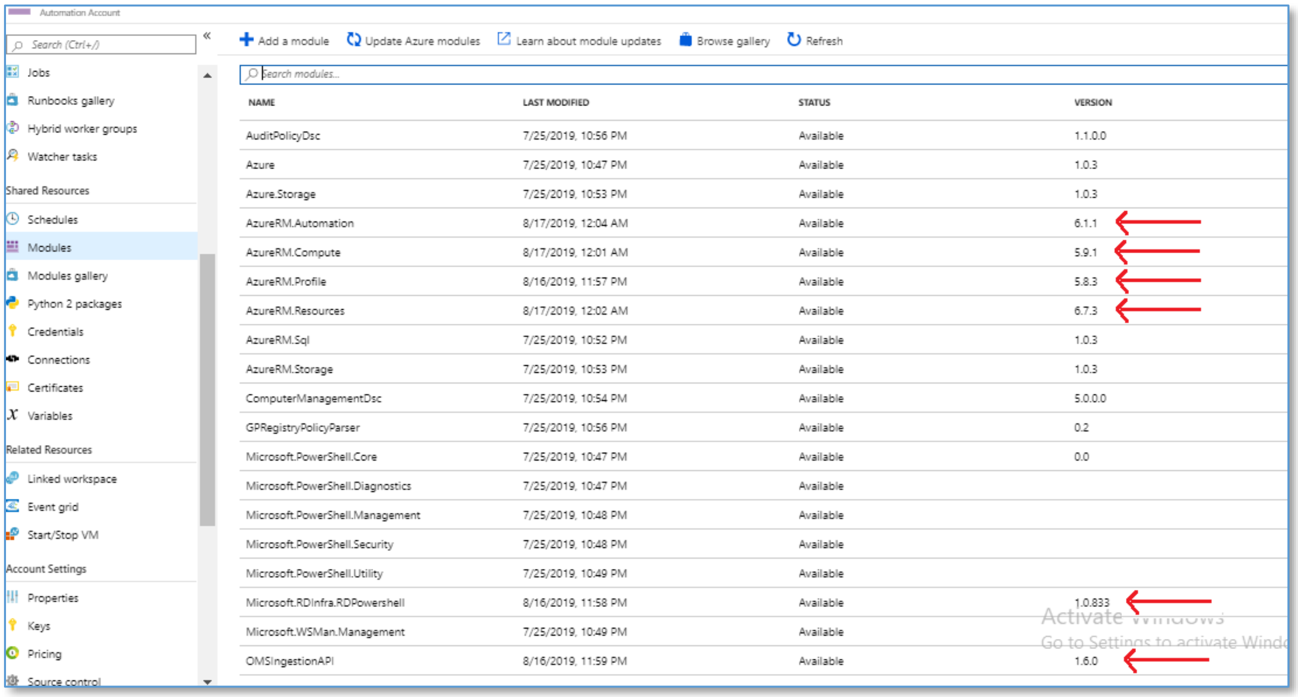
NAME	AUTHORING STATUS	RUNBOOK TYPE	LAST MODIFIED	TAGS
WVDAutoScaleRunbook	✓ Published	PowerShell Runbook	8/8/2019, 2:29 PM	

2. Sets the Service Principal credentials to Automation account Credential Asset.



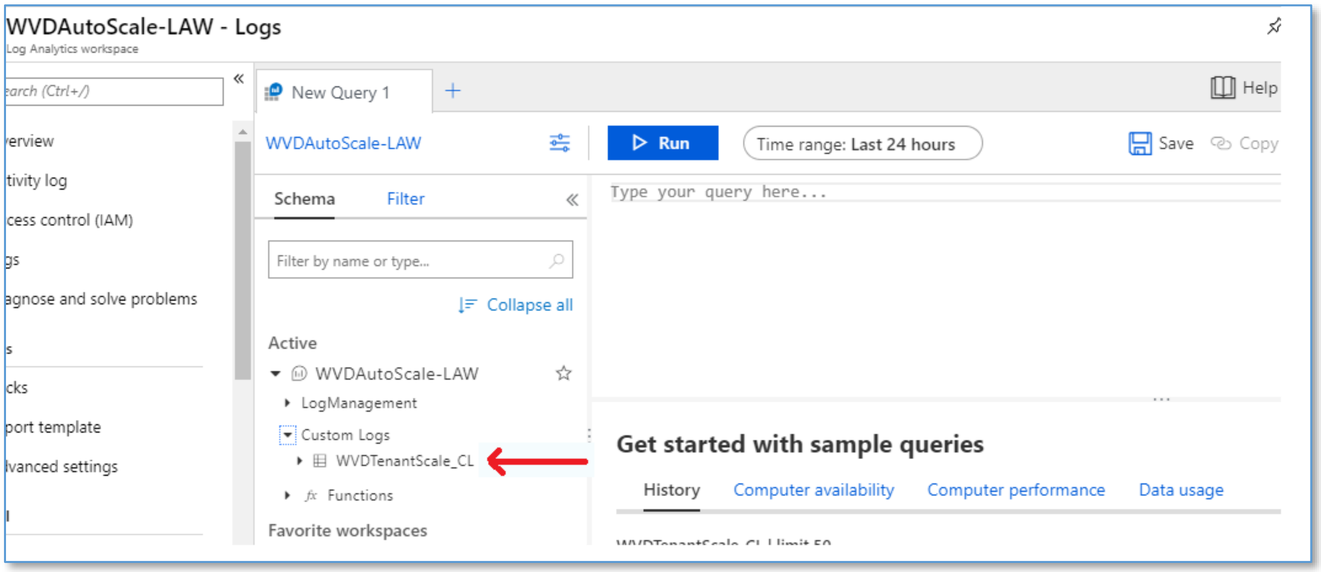
NAME	USER NAME	LAST MODIFIED
WVDAutoScaleSvcPrincipalAsset	95d926fb-5720-45ff-850f-189afa1e8f95	8/8/2019, 1:32 PM

3. Imports below listed modules for executing basic scale script in RunBook
   a. AzureRM.Profile
   b. Microsoft.RDInfra.RDPowershell
   c. OMSIngestionAPI
   d. AzureRM.Compute
   e. AzureRM.Resources
   f. AzureRM.Automation

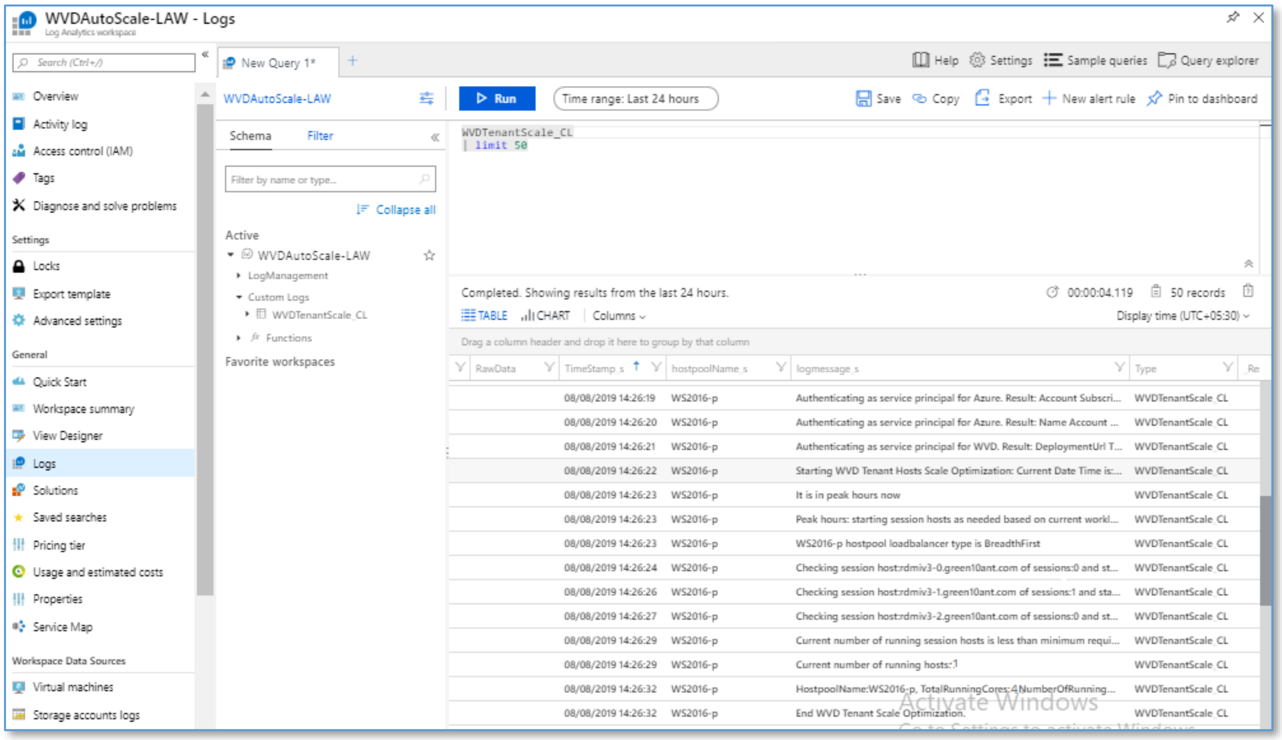


NAME	LAST MODIFIED	STATUS	VERSION
AuditPolicyDsc	7/25/2019, 10:56 PM	Available	1.1.0.0
Azure	7/25/2019, 10:47 PM	Available	1.0.3
Azure.Storage	7/25/2019, 10:53 PM	Available	1.0.3
AzureRM.Automation	8/17/2019, 12:04 AM	Available	6.1.1
AzureRM.Compute	8/17/2019, 12:01 AM	Available	5.9.1
AzureRM.Profile	8/16/2019, 11:57 PM	Available	5.8.3
AzureRM.Resources	8/17/2019, 12:02 AM	Available	6.7.3
AzureRM.Sql	7/25/2019, 10:52 PM	Available	1.0.3
AzureRM.Storage	7/25/2019, 10:53 PM	Available	1.0.3
ComputerManagementDsc	7/25/2019, 10:54 PM	Available	5.0.0.0
GPRegistryPolicyParser	7/25/2019, 10:56 PM	Available	0.2
Microsoft.PowerShell.Core	7/25/2019, 10:47 PM	Available	0.0
Microsoft.PowerShell.Diagnostics	7/25/2019, 10:47 PM	Available	
Microsoft.PowerShell.Management	7/25/2019, 10:48 PM	Available	
Microsoft.PowerShell.Security	7/25/2019, 10:48 PM	Available	
Microsoft.PowerShell.Utility	7/25/2019, 10:49 PM	Available	
Microsoft.RDInfra.RDPowershell	8/16/2019, 11:58 PM	Available	1.0.833
Microsoft.WSMan.Management	7/25/2019, 10:49 PM	Available	
OMSIngestionAPI	8/16/2019, 11:59 PM	Available	1.6.0

4. Creates Custom table (Basic Scale Script Execution Log) in Log Analytics Workspace.



- basicScale.ps1 script execution Logs should be stored as below



```
WVDTenantScale_CL
| limit 50
```

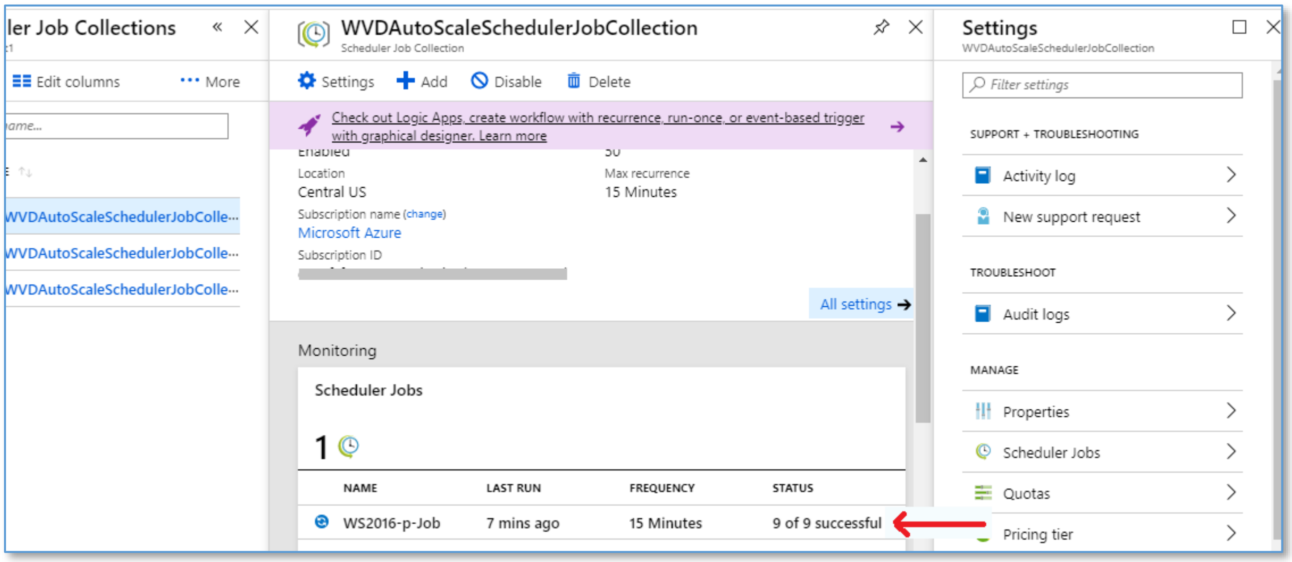
Completed. Showing results from the last 24 hours. 00:00:04.119 50 records

Display time (UTC+05:30)

Drag a column header and drop it here to group by that column

RawData	TimeStamp_s	hostpoolName_s	logmessage_s	Type
	08/08/2019 14:26:19	WS2016-p	Authenticating as service principal for Azure. Result: Account Subscri...	WVDTenantScale_CL
	08/08/2019 14:26:20	WS2016-p	Authenticating as service principal for Azure. Result: Name Account ...	WVDTenantScale_CL
	08/08/2019 14:26:21	WS2016-p	Authenticating as service principal for WVD. Result: DeploymentUrl T...	WVDTenantScale_CL
	08/08/2019 14:26:22	WS2016-p	Starting WVD Tenant Hosts Scale Optimization: Current Date Time is:...	WVDTenantScale_CL
	08/08/2019 14:26:23	WS2016-p	It is in peak hours now	WVDTenantScale_CL
	08/08/2019 14:26:23	WS2016-p	Peak hours: starting session hosts as needed based on current workl...	WVDTenantScale_CL
	08/08/2019 14:26:23	WS2016-p	WS2016-p hostpool loadbalancer type is BreadthFirst	WVDTenantScale_CL
	08/08/2019 14:26:24	WS2016-p	Checking session host:rdmiv3-0.green10ant.com of sessions:0 and st...	WVDTenantScale_CL
	08/08/2019 14:26:26	WS2016-p	Checking session host:rdmiv3-1.green10ant.com of sessions:1 and sta...	WVDTenantScale_CL
	08/08/2019 14:26:27	WS2016-p	Checking session host:rdmiv3-2.green10ant.com of sessions:0 and st...	WVDTenantScale_CL
	08/08/2019 14:26:29	WS2016-p	Current number of running session hosts is less than minimum requi...	WVDTenantScale_CL
	08/08/2019 14:26:29	WS2016-p	Current number of running hosts:1	WVDTenantScale_CL
	08/08/2019 14:26:32	WS2016-p	HostpoolName:WS2016-p, TotalRunningCores:4 NumberOfRunning...	WVDTenantScale_CL
	08/08/2019 14:26:32	WS2016-p	End WVD Tenant Scale Optimization.	WVDTenantScale_CL

5. Deploys the azure scheduler job in Scheduler job collection using **azurScheduler.json** template.


ler Job Collections

Edit columns ··· More

WVDAutoScaleSchedulerJobColle...
WVDAutoScaleSchedulerJobColle...
WVDAutoScaleSchedulerJobColle...

WVDAutoScaleSchedulerJobCollection
Scheduler Job Collection

Settings | Add | Disable | Delete

Check out Logic Apps, create workflow with recurrence, run-once, or event-based trigger with graphical designer. Learn more

Location: Central US
Max recurrence: 15 Minutes
Subscription name (change): Microsoft Azure
Subscription ID: [redacted]

All settings →

Monitoring

Scheduler Jobs

1

NAME	LAST RUN	FREQUENCY	STATUS
WS2016-p-Job	7 mins ago	15 Minutes	9 of 9 successful

Settings
WVDAutoScaleSchedulerJobCollection

Filter settings

SUPPORT + TROUBLESHOOTING
- Activity log
- New support request

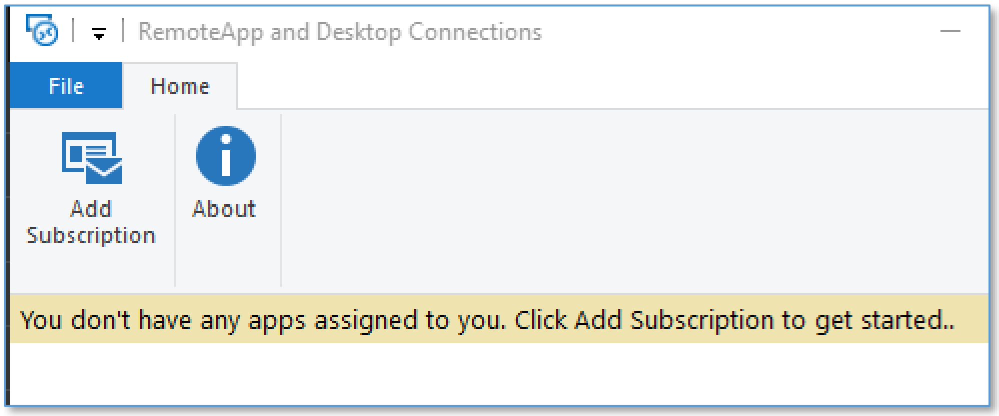
TROUBLESHOOT
- Audit logs

MANAGE
- Properties
- Scheduler Jobs
- Quotas
- Pricing tier

8. Validate user connections to WVD

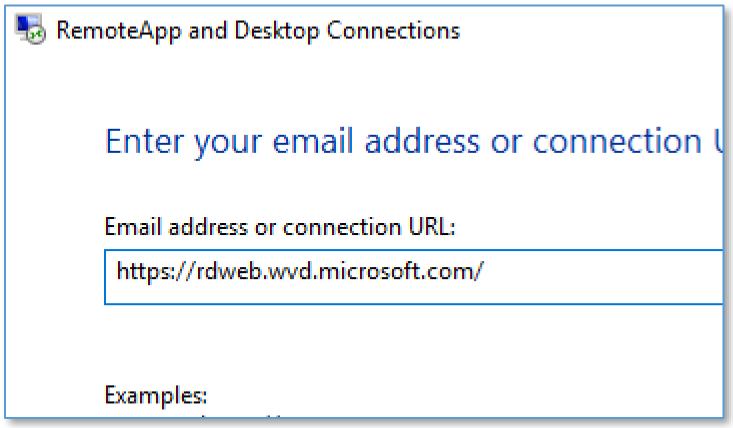
At this stage, your RemoteApps are deployed on the WVD session hosts. A downloadable client is available that provides access to Windows Virtual Desktop resources from devices running Windows 7 and Windows 10. There is a web client that can be used as well.

8.1 Download the client and run the MSI to complete the installation.
8.2 Start the client from the All Apps List, look for Remote Desktop.



You don't have any apps assigned to you. Click Add Subscription to get started..

8.3 Click Add Subscription > provide URL = https://rdweb.wvd.microsoft.com/ > Next > Next Again


RemoteApp and Desktop Connections

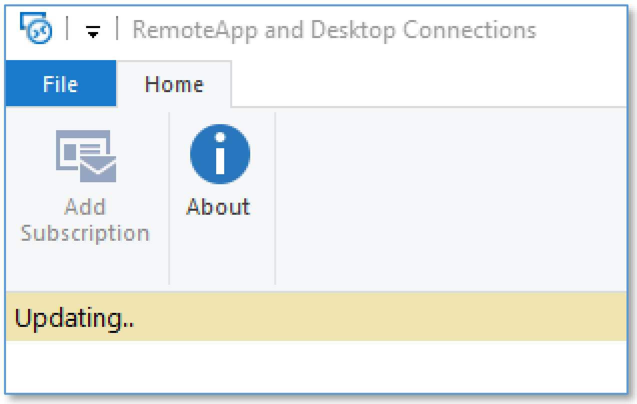
Enter your email address or connection

Email address or connection URL:

https://rdweb.wvd.microsoft.com/

Examples:

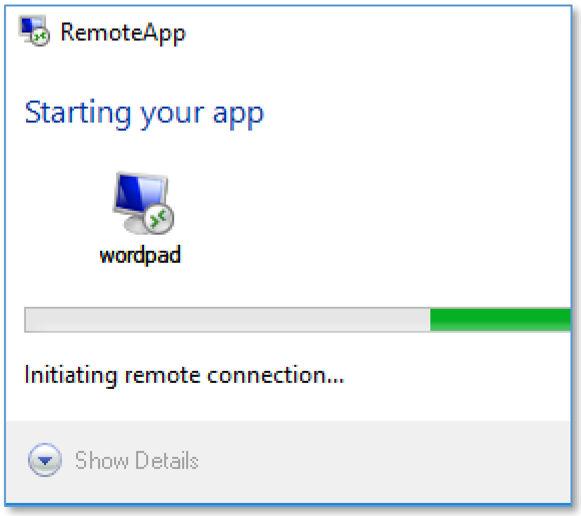
8.4 Sign in with you're the user account that was granted access to the WVD-RemoteApps in the earlier section > Next



Updating..

8.5 After successfully authenticating, you should now see a list of resources available to you.

8.6 Launch any of the resources (EX: WordPad). *please be advised that the first launch may be slow as your user profile is being created.*

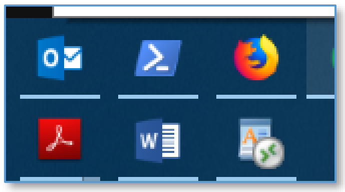

RemoteApp

Starting your app

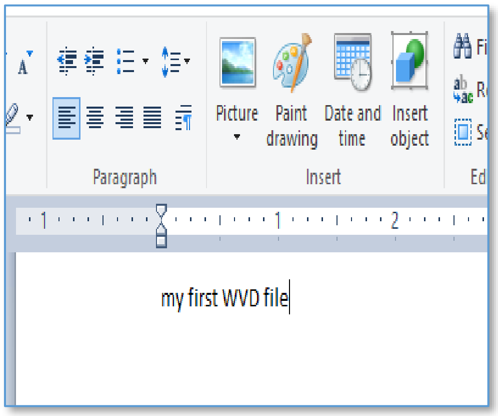
wordpad

Initiating remote connection...

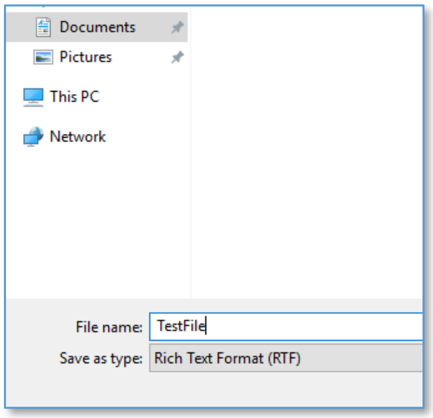
8.7 Once launched, you can see the icon in your taskbar



8.8 Now type something > save your file > close WordPad



my first WVD file



File name: TestFile

Save as type: Rich Text Format (RTF)

8.9 Alternatively, you can have a similar connection experience using a web browser by following the steps below.

NOTE: the browser must be HTML-5 compatible. Supported browsers include latest versions of IE/Edge/Safari/Firefox/Chrome

8.10 Browse to https://rdweb.wvd.microsoft.com
8.11 Login with user domain credentials
8.12 Access Apps & Desktops.
8.13 As an Admin, you can also validate the User Session data from the WVD end using either of the commands:

```
#See users of all AppGroups in a HostPool
Get-RdsUserSession -TenantName $TenantName

#Filter sessions by a specific HostPool
Get-RdsUserSession -TenantName $TenantName -HostPoolName $HostPoolName -Verbose
```

9. VM Host and User Session Management

Set session host to drain mode

The Set-RdsSessionHost cmdlet sets the state of the specified session host. You can either disable or enable new connections to the session host. Changing this property on the session host does not affect any user sessions on the session host.

Run this PowerShell cmdlet to drain a session hosts connections:

```
Set-RdsSessionHost -TenantName "ContosoCorpWVD" -HostPoolName "HPPDW10ENT" -Name
"rdshvm-1.contosocorpwvd.com" -AllowNewSession $false # $true re-enables.
```

Logoff a user

Using the Invoke-RdsUserSessionLogoff cmdlet, you can force a user to logoff. Of course, we would want to send a warning message first to give them a chance to save any work:

```
# Send a message. Update parameters below first:
Get-RdsUserSession -TenantName "ContosoCorpWVD" -HostPoolName "HPPDW10ENT" `
| where { $_.UserPrincipalName -eq "contosocorpwvd\contosouser1" } `
| Send-RdsUserSessionMessage -MessageTitle "Administrative Alert!" `
-MessageBody "Your session will be logged-off in 1 minute. Please save any
work." -NoUserPrompt
```

Wait a minute, then log them Off:

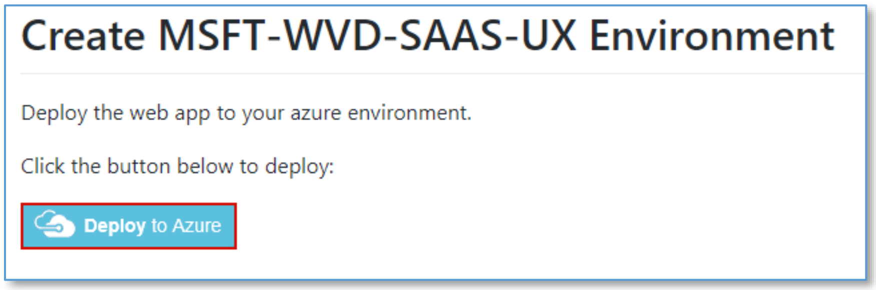
```
# Log them Off. Update parameters below first:
Get-RdsUserSession -TenantName "ContosoCorpWVD" -HostPoolName "HPPDW10ENT" `
| where { $_.UserPrincipalName -eq "contosocorpwvd\contosouser1" } `
| Invoke-RdsUserSessionLogoff -NoUserPrompt
```

10. Deploy the Management UI

Follow the steps below to deploy the WVD management UI and be able to manage some aspects of WVD like provisioning, modifications etc.

10.1 Go to https://github.com/Azure/RDS-Templates/tree/master/wvd-templates/wvd-management-ux/deploy

10.2 Click on **Deploy to Azure:**

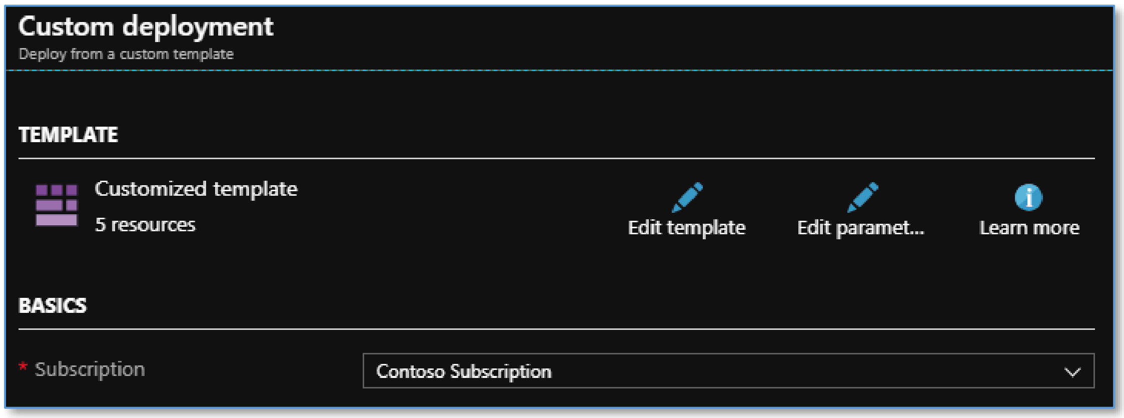


Create MSFT-WVD-SAAS-UX Environment

Deploy the web app to your azure environment.

Click the button below to deploy:

10.3 ON the custom deployment page, provide the following info on the service:



Custom deployment

Deploy from a custom template

TEMPLATE

Customized template

5 resources

BASICS

* Subscription: Contoso Subscription

- RD Broker URL: https://rdbroker.wvd.microsoft.com/
- Resource URL: https://mrs-prod.ame.gbl/mrs-RDInfra-prod
- Azure Login ID/ password – Provide an admin that does not have MFA enforced.
- Input an APP name unique in your subscription, for example "WVDUS2DIAGS"

This will create a resource group and will have 2 app services along with 1 app service plan.

To launch the UX, click on appservice with the name you provided in the template (ex: WVDUS2DIAGS) and navigate to the URL that shows up on the top right side.

NOTE: This works only for admins that have access to Default Tenant Group. We have a fix that needs to be done on the service side for other TGs to work.

11. Diagnose WVD issues with PowerShell

Windows Virtual Desktop Diagnostics uses just one PowerShell cmdlet but contains many optional parameters to help narrow down and isolate issues. In this section we'll access and view the various diagnostics features.

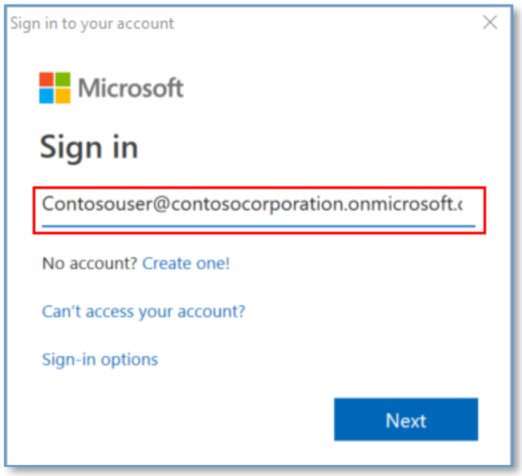
Retrieve diagnostic activities in your tenant

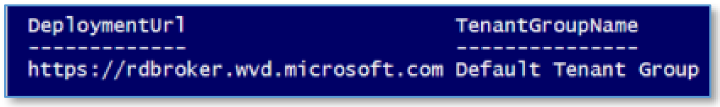
You can retrieve diagnostic activities by using the **Get-RdsDiagnosticActivities** cmdlet. The following example cmdlet will return a list of diagnostic activities, sorted from most to least recent.

From a work station with PowerShell and the Windows Virtual Desktop PowerShell module installed, run the commands below. In our examples, we're using the tenant **ContosoCorpWVD**:

1. Login to the Azure WVD tenant

```
# Import PS module & Logon to the WVD tenant:
Import-Module -Name Microsoft.RDInfra.RDPowerShell
Add-RdsAccount -DeploymentUrl “https://rdbroker.wvd.microsoft.com”
```

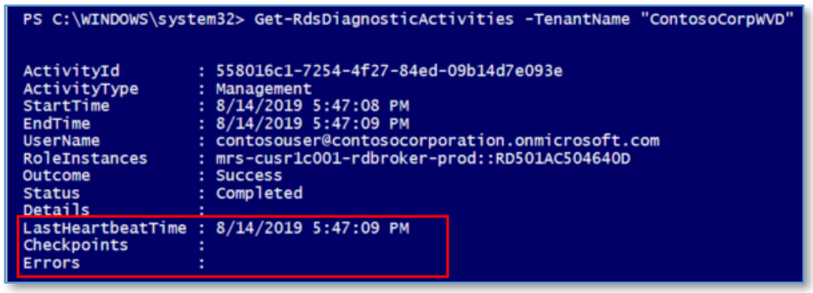




```
DeploymentUrl                     TenantGroupName
-------------                     ---------------
https://rdbroker.wvd.microsoft.com Default Tenant Group
```

2. Show any WVD errors:

```
# Get the diagnostics information:
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD"
```



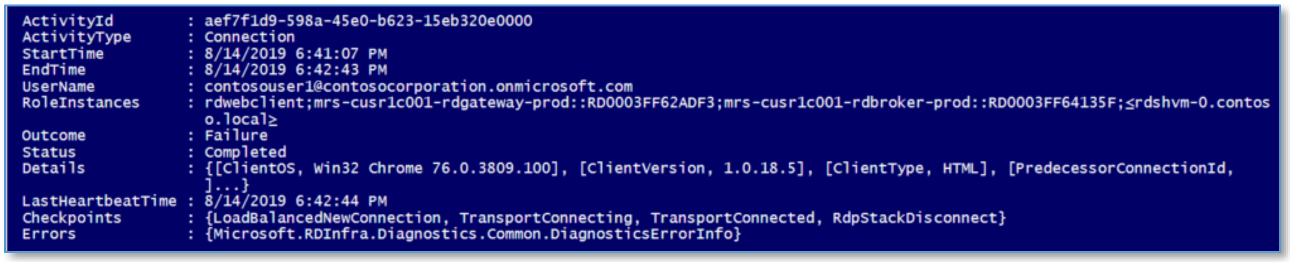
```
PS C:\WINDOWS\system32> Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD"

ActivityId        : 558016c1-7254-4f27-84ed-09b14d7e093e
ActivityType      : Management
StartTime         : 8/14/2019 5:47:08 PM
EndTime           : 8/14/2019 5:47:09 PM
UserName          : contosouser@contosocorporation.onmicrosoft.com
RoleInstances     : mrs-cusr1c001-rdbroker-prod::RD501AC504640D
Outcome           : Success
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 5:47:09 PM
Checkpoints       :
Errors            :
```

That's good news, no errors and **LastHeartBeatTime** is current.

3. Let's create a situation to see the diagnostics feature in action. One way to cause a new diagnostics record, or **Activity**, to be recorded, is by simply ending the WVD session by closing the browser while the user is logged-on. Do this in your own WVD instance, then run the below PowerShell command, using your WVD tenant name:

```
# Get the diagnostics information:
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" -Detailed
```

Ours returns:

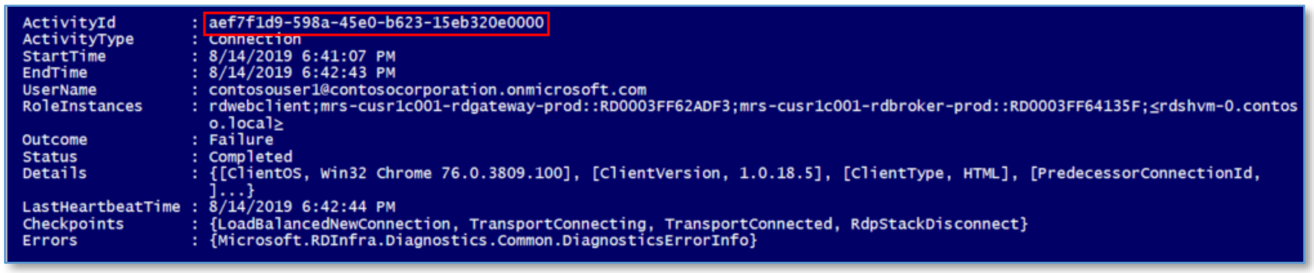


```
ActivityId        : aef7f1d9-598a-45e0-b623-15eb320e0000
ActivityType      : Connection
StartTime         : 8/14/2019 6:41:07 PM
EndTime           : 8/14/2019 6:42:43 PM
UserName          : contosouser1@contosocorporation.onmicrosoft.com
RoleInstances     : rdwebclient;mrs-cusr1c001-rdgateway-prod::RD0003FF62ADF3;mrs-cusr1c001-rdbroker-prod::RD0003FF64135F;≤rdshvm-0.contos
                    o.local≥
Outcome           : Failure
Status            : Completed
Details           : {[ClientOS, Win32 Chrome 76.0.3809.100], [ClientVersion, 1.0.18.5], [ClientType, HTML], [PredecessorConnectionId,
                    ]...}
LastHeartbeatTime : 8/14/2019 6:42:44 PM
Checkpoints       : {LoadBalancedNewConnection, TransportConnecting, TransportConnected, RdpStackDisconnect}
Errors            : {Microsoft.RDInfra.Diagnostics.Common.DiagnosticsErrorInfo}
```

We can see that **Outcome = Failure**, but there's really nothing else useful.

View detailed error messages for a failed activity by activity ID

4. Using the **ActivityID** from the error information:

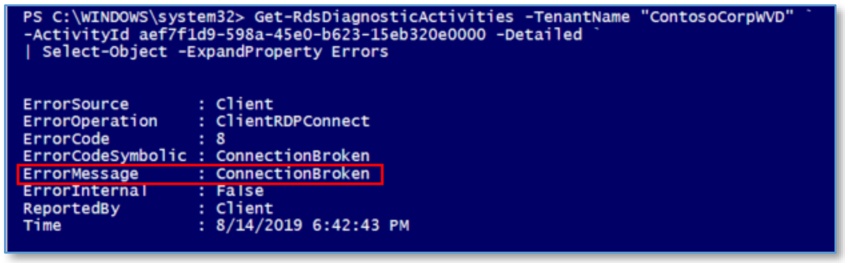


```
ActivityId        : aef7f1d9-598a-45e0-b623-15eb320e0000
ActivityType      : Connection
StartTime         : 8/14/2019 6:41:07 PM
EndTime           : 8/14/2019 6:42:43 PM
UserName          : contosouser1@contosocorporation.onmicrosoft.com
RoleInstances     : rdwebclient;mrs-cusr1c001-rdgateway-prod::RD0003FF62ADF3;mrs-cusr1c001-rdbroker-prod::RD0003FF64135F;≤rdshvm-0.contos
                    o.local≥
Outcome           : Failure
Status            : Completed
Details           : {[ClientOS, Win32 Chrome 76.0.3809.100], [ClientVersion, 1.0.18.5], [ClientType, HTML], [PredecessorConnectionId,
                    ]...}
LastHeartbeatTime : 8/14/2019 6:42:44 PM
Checkpoints       : {LoadBalancedNewConnection, TransportConnecting, TransportConnected, RdpStackDisconnect}
Errors            : {Microsoft.RDInfra.Diagnostics.Common.DiagnosticsErrorInfo}
```

...and adding some additional PowerShell code, we can see more detail:

```
# See errors inner detail using an Activity ID (multiple errors may be listed)
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" `
```

```
-ActivityId aef7f1d9-598a-45e0-b623-15eb320e0000 -Detailed `
| Select-Object -ExpandProperty Errors
```



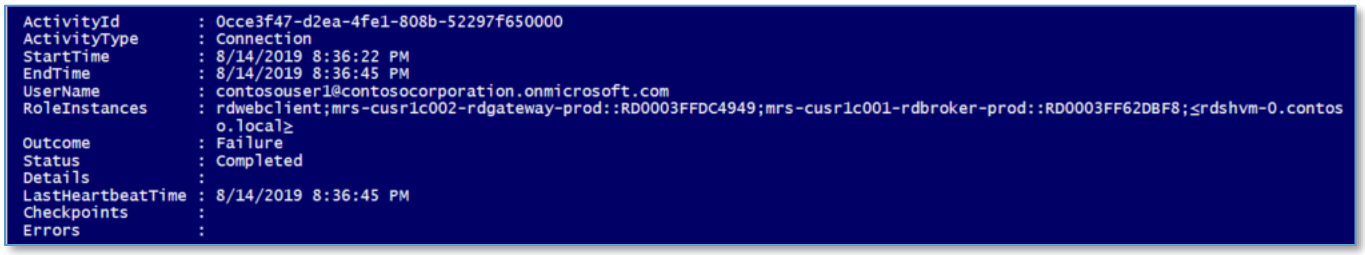
```
PS C:\WINDOWS\system32> Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" `
-ActivityId aef7f1d9-598a-45e0-b623-15eb320e0000 -Detailed `
| Select-Object -ExpandProperty Errors


ErrorSource        : Client
ErrorOperation     : ClientRDPConnect
ErrorCode          : 8
ErrorCodeSymbolic  : ConnectionBroken
ErrorMessage       : ConnectionBroken
ErrorInternal      : False
ReportedBy         : Client
Time               : 8/14/2019 6:42:43 PM
```

Sure enough, the client was disconnected unexpectedly, or not “cleanly”.

Filter diagnostic activities by user

Using the **-UserName** parameter allows us to filter all of the diagnostics information by that users` logon ID. In our example below, we’ll locate specific error information for our Contosouser1 user:

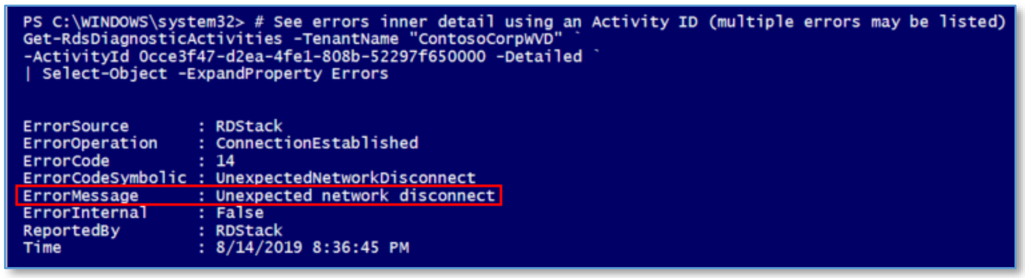
```
# Filter errors by user
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" `
-UserName contosouser1@contosocorporation.onmicrosoft.com
```



```
ActivityId         : 0cce3f47-d2ea-4fe1-808b-52297f650000
ActivityType       : Connection
StartTime          : 8/14/2019 8:36:22 PM
EndTime            : 8/14/2019 8:36:45 PM
UserName           : contosouser1@contosocorporation.onmicrosoft.com
RoleInstances      : rdwebclient;mrs-cusr1c002-rdgateway-prod::RD0003FFDC4949;mrs-cusr1c001-rdbroker-prod::RD0003FF62DBF8;<rdshvm-0.contos
                     o.local>
Outcome            : Failure
Status             : Completed
Details            :
LastHeartbeatTime  : 8/14/2019 8:36:45 PM
Checkpoints        :
Errors             :
```

Using the **ActivityID** above, and the PS code from earlier, let’s see the detailed error information:

```
# See errors inner detail using an Activity ID (multiple errors may be listed)
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" `
-ActivityId 0cce3f47-d2ea-4fe1-808b-52297f650000 -Detailed `
| Select-Object -ExpandProperty Errors
```



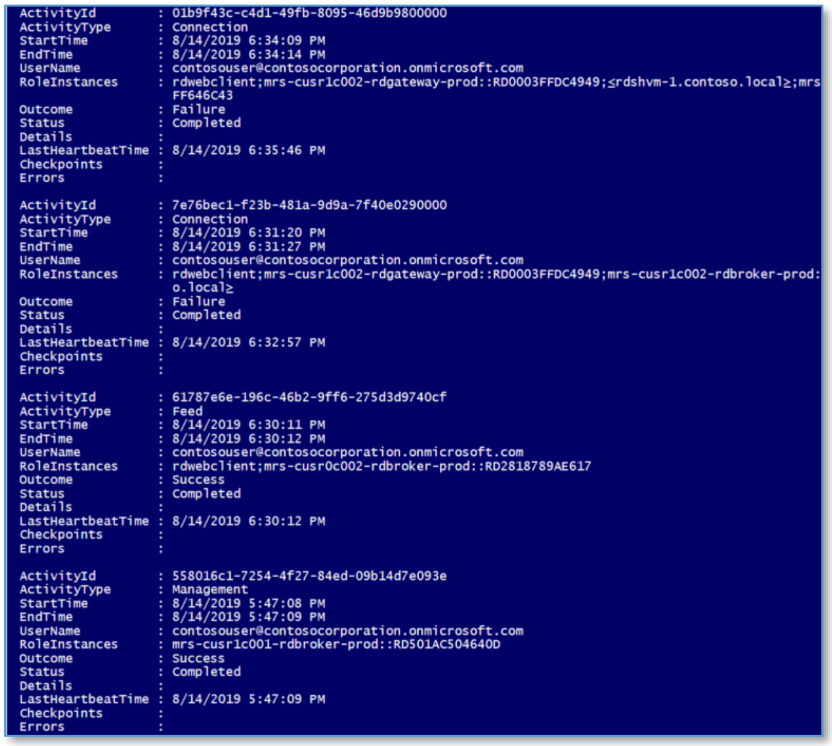
```
PS C:\WINDOWS\system32> # See errors inner detail using an Activity ID (multiple errors may be listed)
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" `
-ActivityId 0cce3f47-d2ea-4fe1-808b-52297f650000 -Detailed `
| Select-Object -ExpandProperty Errors


ErrorSource        : RDStack
ErrorOperation     : ConnectionEstablished
ErrorCode          : 14
ErrorCodeSymbolic  : UnexpectedNetworkDisconnect
ErrorMessage       : Unexpected network disconnect
ErrorInternal      : False
ReportedBy         : RDStack
Time               : 8/14/2019 8:36:45 PM
```

Once again, we can see that the client was disconnected unexpectedly.

Filter diagnostic activities by date

You can filter the returned diagnostic activity list with the -**StartTime** and -**EndTime** parameters. The -StartTime parameter used alone, will return a diagnostic activity list starting from a specific date, as shown in the following example.

```
# Filter activities by date
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" -StartTime "08/14/2019"
```

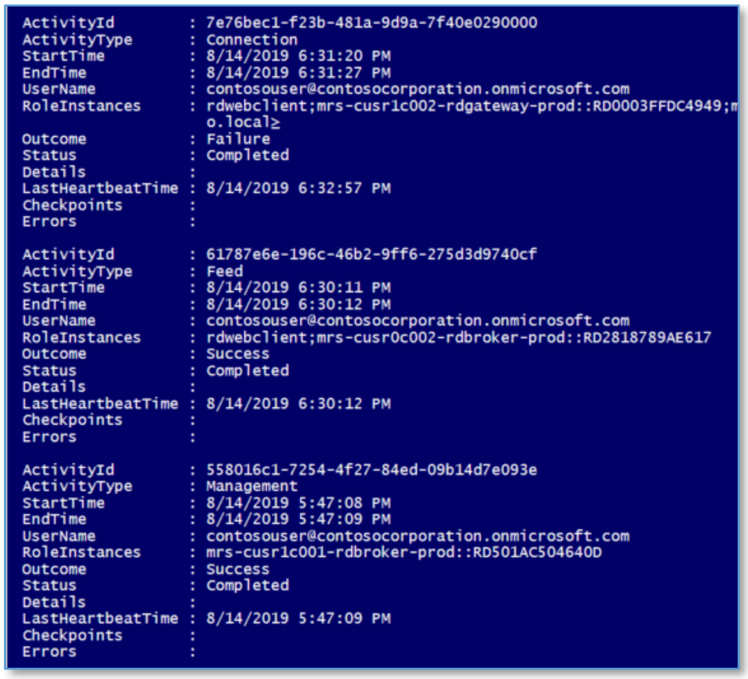


```
ActivityId        : 01b9f43c-c4d1-49fb-8095-46d9b9800000
ActivityType      : Connection
StartTime         : 8/14/2019 6:34:09 PM
EndTime           : 8/14/2019 6:34:14 PM
UserName          : contosouser@contosocorporation.onmicrosoft.com
RoleInstances     : rdwebclient;mrs-cusr1c002-rdgateway-prod::RD0003FFDC4949;≤rdshvm-1.contoso.local≥;mrs
                    FF646C43
Outcome           : Failure
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 6:35:46 PM
Checkpoints       :
Errors            :

ActivityId        : 7e76bec1-f23b-481a-9d9a-7f40e0290000
ActivityType      : Connection
StartTime         : 8/14/2019 6:31:20 PM
EndTime           : 8/14/2019 6:31:27 PM
UserName          : contosouser@contosocorporation.onmicrosoft.com
RoleInstances     : rdwebclient;mrs-cusr1c002-rdgateway-prod::RD0003FFDC4949;mrs-cusr1c002-rdbroker-prod:
                    o.local≥
Outcome           : Failure
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 6:32:57 PM
Checkpoints       :
Errors            :

ActivityId        : 61787e6e-196c-46b2-9ff6-275d3d9740cf
ActivityType      : Feed
StartTime         : 8/14/2019 6:30:11 PM
EndTime           : 8/14/2019 6:30:12 PM
UserName          : contosouser@contosocorporation.onmicrosoft.com
RoleInstances     : rdwebclient;mrs-cusr0c002-rdbroker-prod::RD2818789AE617
Outcome           : Success
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 6:30:12 PM
Checkpoints       :
Errors            :

ActivityId        : 558016c1-7254-4f27-84ed-09b14d7e093e
ActivityType      : Management
StartTime         : 8/14/2019 5:47:08 PM
EndTime           : 8/14/2019 5:47:09 PM
UserName          : contosouser@contosocorporation.onmicrosoft.com
RoleInstances     : mrs-cusr1c001-rdbroker-prod::RD501AC504640D
Outcome           : Success
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 5:47:09 PM
Checkpoints       :
Errors            :
```

... *and on and on.*

Add the -**EndTime** parameter to limit the returned list size:

```
# Filter activities by date
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" `
-StartTime "8/14/2019 5:47:08 PM" -EndTime "8/14/2019 6:32:00 PM"
```



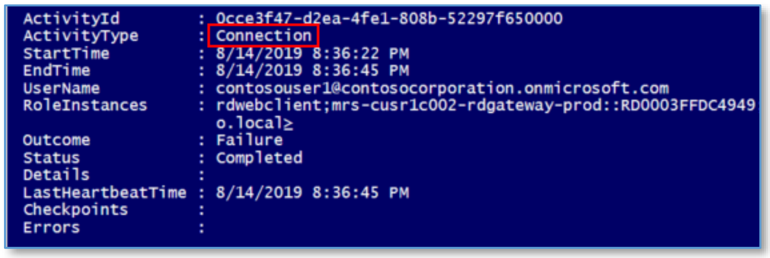
```
ActivityId        : 7e76bec1-f23b-481a-9d9a-7f40e0290000
ActivityType      : Connection
StartTime         : 8/14/2019 6:31:20 PM
EndTime           : 8/14/2019 6:31:27 PM
UserName          : contosouser@contosocorporation.onmicrosoft.com
RoleInstances     : rdwebclient;mrs-cusr1c002-rdgateway-prod::RD0003FFDC4949;m
                    o.local≥
Outcome           : Failure
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 6:32:57 PM
Checkpoints       :
Errors            :

ActivityId        : 61787e6e-196c-46b2-9ff6-275d3d9740cf
ActivityType      : Feed
StartTime         : 8/14/2019 6:30:11 PM
EndTime           : 8/14/2019 6:30:12 PM
UserName          : contosouser@contosocorporation.onmicrosoft.com
RoleInstances     : rdwebclient;mrs-cusr0c002-rdbroker-prod::RD2818789AE617
Outcome           : Success
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 6:30:12 PM
Checkpoints       :
Errors            :

ActivityId        : 558016c1-7254-4f27-84ed-09b14d7e093e
ActivityType      : Management
StartTime         : 8/14/2019 5:47:08 PM
EndTime           : 8/14/2019 5:47:09 PM
UserName          : contosouser@contosocorporation.onmicrosoft.com
RoleInstances     : mrs-cusr1c001-rdbroker-prod::RD501AC504640D
Outcome           : Success
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 5:47:09 PM
Checkpoints       :
Errors            :
```

This time, the returned list is limited to the 3 records we're interested in.

Filter diagnostic activities by activity type

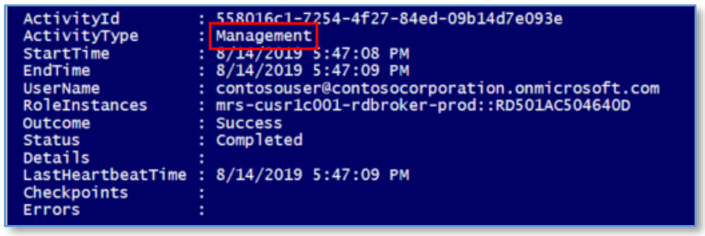
Using the **-ActivityType** parameter, we can see Connection and Management related activities:

```
# Filter activities by Connection ActivityType
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" `
-ActivityType Connection
```



```
ActivityId        : 0cce3f47-d2ea-4fe1-808b-52297f650000
ActivityType      : Connection
StartTime         : 8/14/2019 8:36:22 PM
EndTime           : 8/14/2019 8:36:45 PM
UserName          : contosouser1@contosocorporation.onmicrosoft.com
RoleInstances     : rdwebclient;mrs-cusr1c002-rdgateway-prod::RD0003FFDC4949
                    o.local≥
Outcome           : Failure
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 8:36:45 PM
Checkpoints       :
Errors            :
```

```
# Filter activities by Management ActivityType
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" `
-ActivityType Management
```

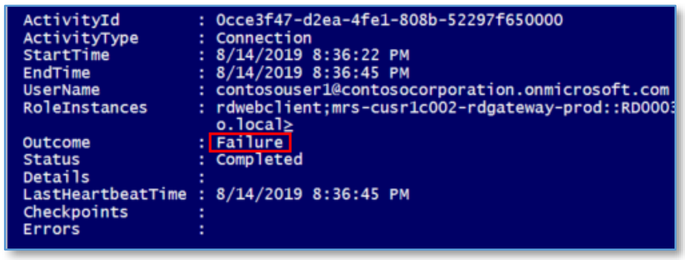


```
ActivityId        : 558016c1-7254-4f27-84ed-09b14d7e093e
ActivityType      : Management
StartTime         : 8/14/2019 5:47:08 PM
EndTime           : 8/14/2019 5:47:09 PM
UserName          : contosouser@contosocorporation.onmicrosoft.com
RoleInstances     : mrs-cusr1c001-rdbroker-prod::RD501AC504640D
Outcome           : Success
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 5:47:09 PM
Checkpoints       :
Errors            :
```

Filter diagnostic activities by outcome

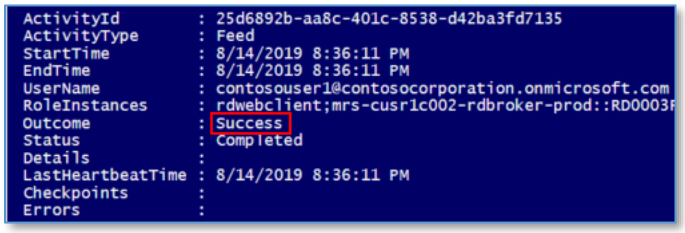
Use the -**Outcome** parameter to find diagnostic activities that were either successful or have failed:

```
# Filter activities by failed Outcome
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" `
-Outcome Failure
```



```
ActivityId        : 0cce3f47-d2ea-4fe1-808b-52297f650000
ActivityType      : Connection
StartTime         : 8/14/2019 8:36:22 PM
EndTime           : 8/14/2019 8:36:45 PM
UserName          : contosouser1@contosocorporation.onmicrosoft.com
RoleInstances     : rdwebclient;mrs-cusr1c002-rdgateway-prod::RD0003
                    o.local>
Outcome           : Failure
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 8:36:45 PM
Checkpoints       :
Errors            :
```

```
# Filter activities by successful Outcome
Get-RdsDiagnosticActivities -TenantName "ContosoCorpWVD" `
-Outcome Success
```



```
ActivityId        : 25d6892b-aa8c-401c-8538-d42ba3fd7135
ActivityType      : Feed
StartTime         : 8/14/2019 8:36:11 PM
EndTime           : 8/14/2019 8:36:11 PM
UserName          : contosouser1@contosocorporation.onmicrosoft.com
RoleInstances     : rdwebclient;mrs-cusr1c002-rdbroker-prod::RD0003F
Outcome           : Success
Status            : Completed
Details           :
LastHeartbeatTime : 8/14/2019 8:36:11 PM
Checkpoints       :
Errors            :
```

The -**Outcome** parameter can also be combined with other optional filtering parameters.

12. Support

Opening tickets

In case of an issue for Windows Virtual Desktop go to the Azure Portal and open a technical ticket based on your existing support plan at https://azure.microsoft.com/en-us/support/create-ticket/

Look for Service under **COMPUTE** and select **Windows Virtual Desktop-Preview**. You will find options to create tickets for the WVD service itself and for Office:

For Office issues you can file tickets during public preview in the Azure Portal when using Office in context of Windows Virtual Desktop.

Information you should provide for failed connection or management interactions when using the service:

- Use the diagnostics service to retrieve the **Activity ID** for failed connections or management interactions.
- Provide the approximate timeframe the issue happened

NOTE: This workflow will change post general availability.

Other resources you can leverage

Windows Virtual Desktop contains a number of knowledge articles as well as trouble shooting guides. Pay attention to the updated diagnostics chapter that provides Error scenarios you can mitigate: https://docs.microsoft.com/azure/virtual-desktop/overview

Exchange on our community forum on issues important to you for Windows Virtual Desktop: https://techcommunity.microsoft.com/t5/Windows-Virtual-Desktop/bd-p/WindowsVirtualDesktop

When setting up your environment you will be using other Azure Services. You can watch the health dashboard here to verify health state on any Azure service you are consuming: https://azure.microsoft.com/en-us/status/