# Automatically scale session hosts using Azure Automation

## Introduction

# For many Windows Virtual Desktop deployments in Azure, the virtual machine costs represent sizable portion of the total Windows Virtual Desktop deployment cost. To reduce costs, it is best to shut down and deallocate session host virtual machines (VMs) during off-peak usage hours, then restart them during peak usage hours.

This article describes a way to use Azure Automation to automatically scale session host virtual machines in your Windows Virtual Desktop environment. To learn more about how the scaling tool works, see the How the scaling tool works section.

## Prerequisites

Below are the prerequisites for this implementation:

* [Windows Virtual Desktop tenant and host pool created](https://docs.microsoft.com/en-us/azure/virtual-desktop/create-host-pools-arm-template)
* Session host pool VMs configured and registered with the Windows Virtual Desktop service
* Logic app to be created in the same subscription where the Pools/VMs reside
* A user with [Contributor access](https://docs.microsoft.com/en-us/azure/role-based-access-control/role-assignments-portal) on Azure subscription
* Powershell requirements on machine from where you plan to carry out the deployment:
  1. [Windows PowerShell 5.1](https://docs.microsoft.com/en-us/powershell/scripting/install/installing-windows-powershell" \l "upgrading-existing-windows-powershell)
  2. [Microsoft Az PowerShell module](https://docs.microsoft.com/en-us/powershell/azure/install-az-ps?view=azps-3.1.0)

## Deployment

There are three main steps to setup the Scaling tool:

1. Create Azure Automation account and publish Powershell runbook
2. Create Azure Run As account for accessing Windows Virtual Desktop and Azure resources
3. Create Azure Logic App and execution schedule for the scaling tool

### Create Azure Automation account to run Powershell runbook

1. Login to the machine from where you plan to carry out the deployment
2. Open Windows PowerShell as an Administrator
3. Run the following cmdlet to sign-in to the Azure Account. Your account needs to have contributor rights on Azure subscription that you would like to use for deploying the scaling tool

Login-AzAccount

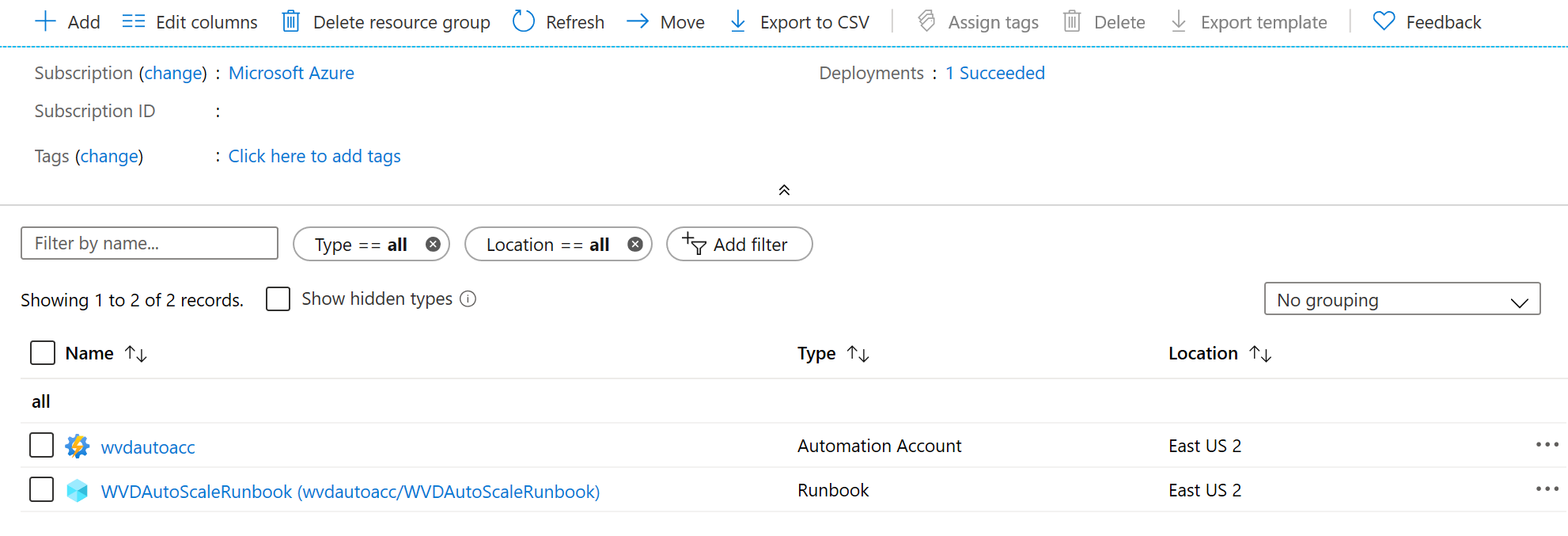
1. Run the following cmdlet to download the script for creating the automation account:

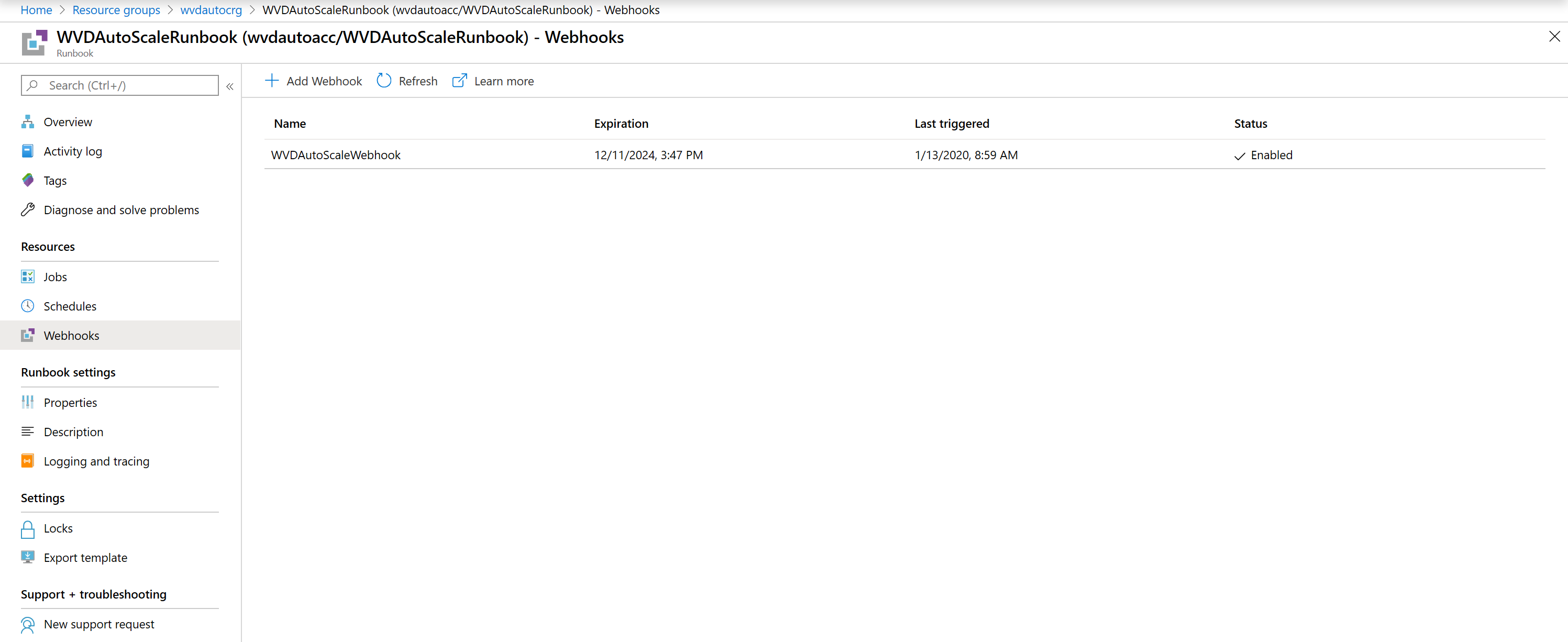
Invoke-WebRequest -Uri “https://raw.githubusercontent.com/Azure/RDS-Templates/ptg-wvdautoscaling-automation/wvd-templates/wvd-scaling-script/wvdscaling-automation/createazureautomationaccount.ps1" -OutFile “your local machine path\ createazureautomationaccount.ps1”

1. Run the following command to execute the script and create the Azure Automation Account:

.\createazureautomationaccount.ps1 -SubscriptionID <azuresubscriptionid> -ResourceGroupName <resourcegroupname> –AutomationAccountName <name of automation account> -Location "Azure region for deployment"

1. Take note of Webhook URI from output of above command as it will be used as parameter while setting up an execution schedule with logic apps.
2. On completion of these steps following resources are created in specified resource groups:
   1. Automation Account
   2. Runbook
   3. Webhook





### Create Azure Automation Run As account for accessing Windows Virtual Desktop and Azure resources

Azure Run As accounts in Azure Automation are used to provide authentication for managing resources in Azure with the Azure cmdlets. When you create a Run As account, it creates a new service principal user in Azure Active Directory and assigns the Contributor role to this user at the subscription level, the Azure Run As Account is a great way to authenticate securely with certificates and a service principal name without needing to store a username and password in a credential object. More details on managing Azure Automation Run As accounts can be found [here](https://aka.ms/limitrunaspermission).

These steps can be performed by a different persona that is a member of the Subscription Admins role and co-administrator of the subscription.

#### Perform the following steps to update your Azure Automation account in the Azure portal

1. In the Azure portal, click All services. In the list of resources, type Automation. As you begin typing, the list filters based on your input. Select **Automation Accounts**.

3. On the **Automation Accounts** page, select your Automation account from the list of Automation accounts.

4. In the left-hand pane, select **Run As Accounts** under the section **Account Settings**.

5. Select **Azure Run As Account**. After selecting the same , **Add Azure Run As Account** pane appears and after reviewing the overview information, click **Create** to proceed with Run As account creation.

6. While Azure creates the Run As account, you can track the progress under Notifications from the menu. A banner is also displayed stating the account is being created. This process can take a few minutes to complete.

7. This creates an Automation connection asset named AzureRunAsConnection in the specified Automation account. The connection asset holds the applicationId, tenantId, subscriptionId, and certificate thumbprint.

8. Take note of applicationid and connection asset name

#### Create a role assignment in Windows Virtual Desktop

Next, you need to create a role assignment so that AzureRunAsConnection can interact with Windows Virtual Desktop. Make sure to sign in with an account that has permissions to create role assignments.

First, download and import the [Windows Virtual Desktop PowerShell module](https://docs.microsoft.com/powershell/windows-virtual-desktop/overview) to use in your PowerShell session if you haven't already. Run the following PowerShell cmdlets to connect to Windows Virtual Desktop and display your tenants.

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

Get-RdsTenant

When you find the tenant you want to scale, use that name in the following cmdlet and the application ID from the [Perform the following steps to update your Azure Automation account in the Azure portal](#_Perform_the_following) steps to create the role assignment:

New-RdsRoleAssignment -RoleDefinitionName "RDS Contributor" -ApplicationId <applicationid> -TenantName <tenantname>

### Create Azure Logic App and execution schedule for the scaling tool

1. Open Windows PowerShell as an Administrator
2. Run the following cmdlet to sign-in to the Azure Account.

Login-AzAccount

1. Run the following cmdlet to download “createazurelogicapp.ps1” script file into your local machine.

Invoke-WebRequest -Uri “https://raw.githubusercontent.com/Azure/RDS-Templates/ptg-wvdautoscaling-automation/wvd-templates/wvd-scaling-script/wvdscaling-automation/createazurelogicapp.ps1" -OutFile “your local machine path\ createazurelogicapp.ps1”

1. Run the following cmdlet to sign into Windows Virtual Desktop with account which has RDS Owner or RDS Contributor permissions

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

1. Execute the following script to create logic app and execution schedule

$aadTenantId = Read-Host -Prompt “Enter your Azure AD tenant ID”

$subscriptionId = Read-Host -Prompt “Enter your Azure Subscription ID”

$tenantName = Read-Host -Prompt “Enter the name of your WVD tenant”

$hostPoolName = Read-Host -Prompt “Enter the name of the host pool you’d like to scale”

$recurrenceInterval = Read-Host -Prompt “Enter how often you’d like the job to run in minutes, e.g. ‘15’”

$beginPeakTime = Read-Host -Prompt “Enter the start time for peak hours in local time, e.g. 9:00”

$endPeakTime = Read-Host -Prompt “Enter the end time for peak hours in local time, e.g. 18:00”

$timeDifference = Read-Host -Prompt “Enter the time difference between local time and UTC in hours, e.g. +5:30”

$sessionThresholdPerCPU = Read-Host -Prompt “Enter the maximum number of sessions per CPU that will be used as a threshold to determine when new session host VMs need to be started during peak hours”

$minimumNumberOfRdsh = Read-Host -Prompt “Enter the minimum number of session host VMs to keep running during off-peak hours”

$limitSecondsToForceLogOffUser = Read-Host -Prompt “Enter the number of seconds to wait before automatically signing out users. If set to 0, users will be signed out immediately”

$logOffMessageTitle = Read-Host -Prompt “Enter the title of the message sent to the user before they are forced to sign out”

$logOffMessageBody = Read-Host -Prompt “Enter the body of the message sent to the user before they are forced to sign out”

$location = Read-Host -Prompt “Enter the name of the Azure region where you will be creating the logic app”

$connectionAssetName = Read-Host -Prompt “Enter the name of the Azure RunAs connection asset”

$webHookURI = Read-Host -Prompt “Enter the URI of the WebHook returned by when you created the Azure Automation Account”

$automationAccountName = Read-Host -Prompt “Enter the name of the Azure Automation Account”

$maintenanceTagName = Read-Host -Prompt “Enter the name of the Tag associated with VMs you don’t want to be managed by this scaling tool”

.\createazurelogicapp.ps1 -ResourceGroupName “Name of the resource group” `

-AADTenantID $aadTenantId `

-SubscriptionID $subscriptionId `

-TenantName $tenantName `

-HostPoolName $hostPoolName `

-RecurrenceInterval $recurrenceInterval `

-BeginPeakTime $beginPeakTime `

-EndPeakTime $endPeakTime `

-TimeDifference $timeDifference `

-SessionThresholdPerCPU $sessionThresholdPerCPU `

-MinimumNumberOfRDSH $minimumNumberOfRdsh `

-LimitSecondsToForceLogOffUser $limitSecondsToForceLogOffUser `

-LogOffMessageTitle $logOffMessageTitle `

-LogOffMessageBody $logOffMessageBody `

-Location $location `

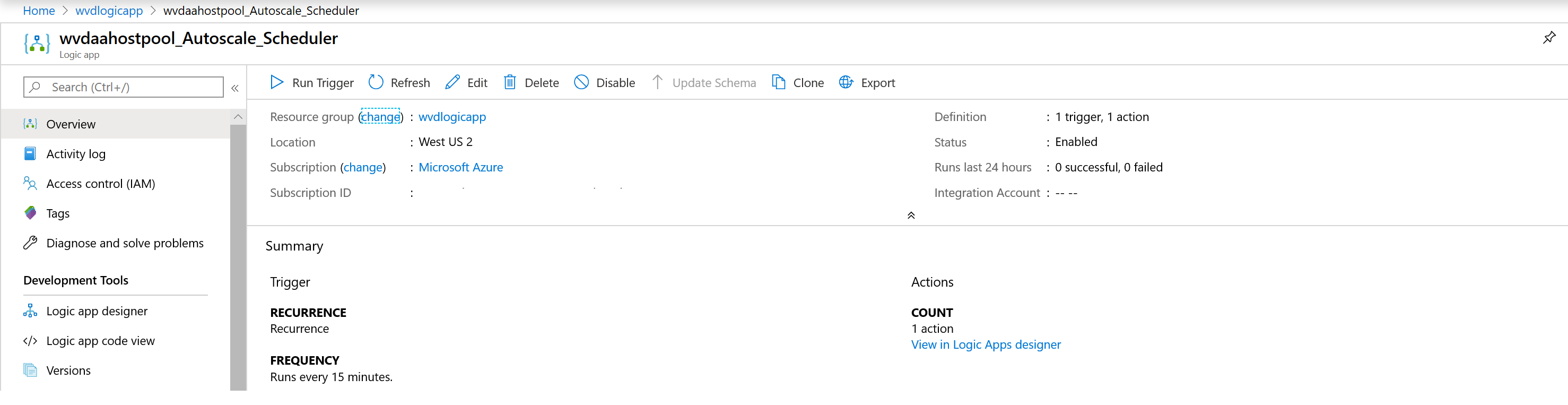
-ConnectionAssetName $connectionAssetName `

-WebHookURI $webHookURI `

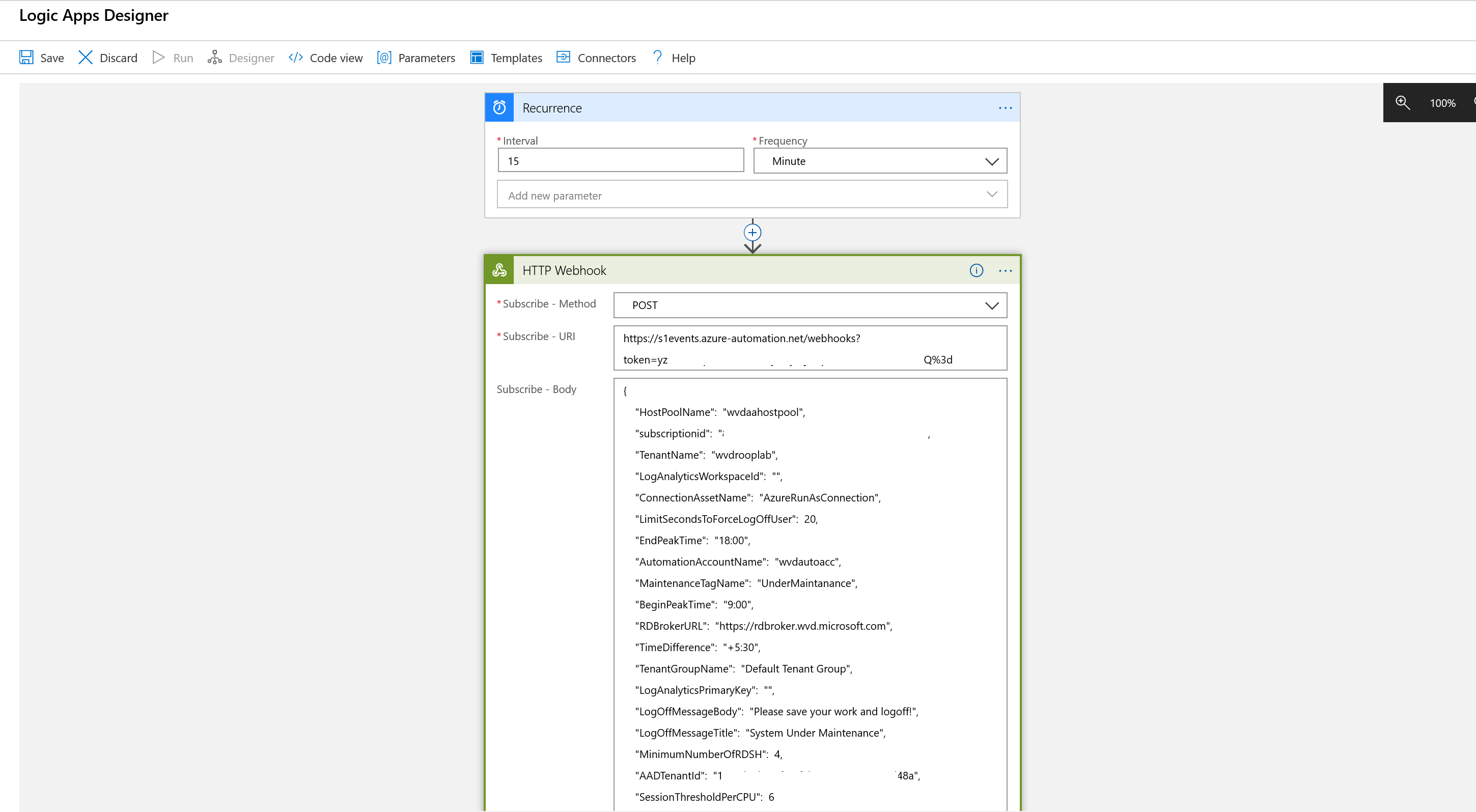
-AutomationAccountName $automationAccountName `

-MaintenanceTagName $maintenanceTagName

1. On completion of this step, logic app with execution schedule gets created and can be viewed by going to resource group hosting the Logic App



Any future changes in execution schedule like recurrence interval or time zone can be updated in Logic Apps Designer which can be accessed by clicking Edit option in Autoscale scheduler.



## Manage your environment

### View status of the jobs

You can view a summarized status of all runbook jobs or drill into details of a specific runbook job in the Azure portal.

On the right of your selected Automation account, you can see a summary of all the runbook jobs under Job Statistics tile.

A picture containing screenshot

Description automatically generated

Clicking on the Jobs page on the left shows the status, start times, and completion times.

A screenshot of a cell phone

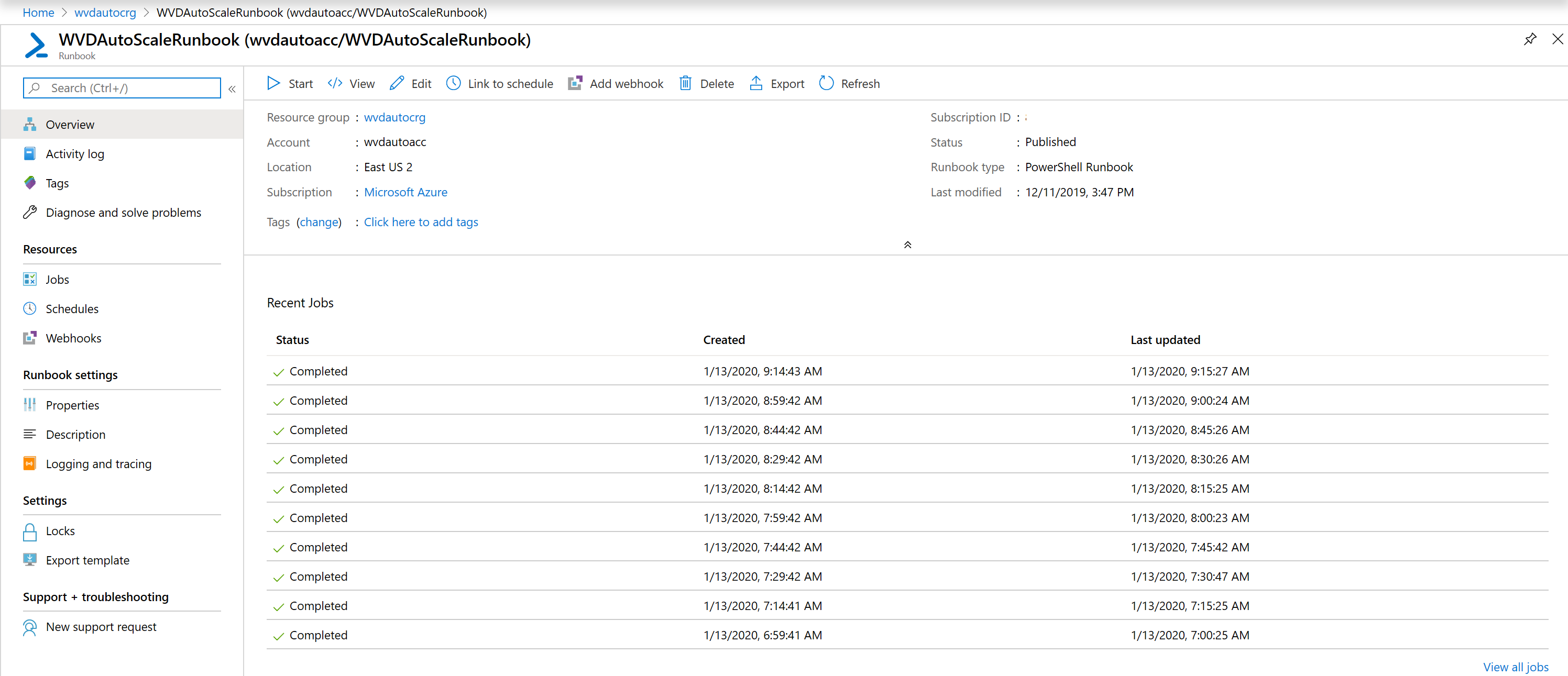
Description automatically generated

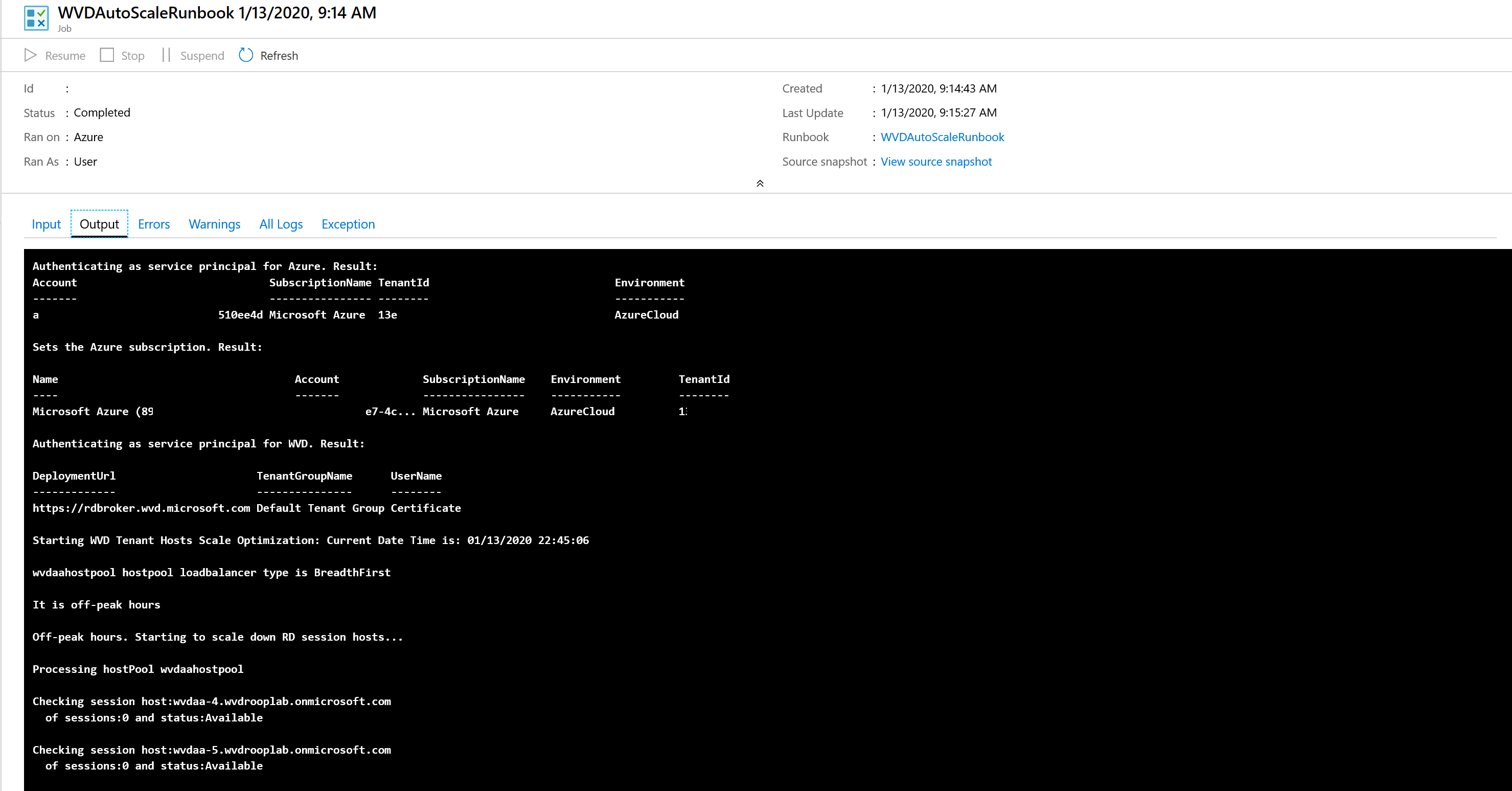
### View output of the scaling tool during Peak and Off-Peak hours

Logs of Scale-out and Scale-in operations can be found by clicking on the job under the Runbook.

Navigate to the WVDAutoScaleRunbook (default name of the RunBook) in your resource group hosting the Azure Automation account

Under Overview section, click on any of the Recent Jobs to view output of scaling tool.





## How the scaling tool works

Scaling tool has been implemented using Azure Automation Runbook, Webhook and an Azure Logic Apps. Azure Logic Apps calls the webhook to start the Azure Automation runbook which implements the scaling function. When you start a Runbook in Azure Automation, a job is created. A job is a single execution instance of a runbook.

Automation job gathers it settings from inputs provided to the **createazurelogicapp.ps1** file, including the start and end of the peak usage period during the day.

During peak usage time, the job checks the current number of sessions and the current running session host VM capacity for each host pool. It calculates if the running session host VMs have enough capacity to support existing sessions based on the SessionThresholdPerCPU parameter defined for the **createazurelogicapp.ps1** file. If not, the job starts additional session host VMs in the host pool.

During the off-peak usage time, the job determines which session host VMs should shut down based on the MinimumNumberOfRDSH parameter. The job will set the session host VMs to drain mode to prevent new sessions connecting to the hosts. If you set the LimitSecondsToForceLogOffUser parameter to a non-zero positive value, the script will notify any currently signed in users to save work, wait the configured amount of time, and then force the users to sign out. Once all user sessions have been signed off on a session host VM, the script will shut down the server.

If you set the LimitSecondsToForceLogOffUser parameter to zero, the job will allow the session configuration setting in the host pool properties to handle signing off user sessions. If there are any sessions on a session host VM, it will leave the session host VM running. If there aren't any sessions, the job will shut down the session host VM.

The job is designed to run periodically based on set recurrence interval. Select the appropriate time interval based on the size of your Windows Virtual Desktop environment and remember that starting and shutting down virtual machines can take some time. We recommend setting recurrence interval every 15 minutes.

Note: Scaling tool controls the load balancing mode of the host pool it is scaling. It sets it to BreadthFirst load balancing for both Peak and Off-Peak hours.