# Azure Virtual Desktop x Shutdown and deallocate Session host at logoff

Table of contents

1. [Introduction](https://avdpunks.com/2023/01/12/avd-shutdown-and-deallocate-session-host-at-logoff/#Introduction)
2. [Components and workflow](https://avdpunks.com/2023/01/12/avd-shutdown-and-deallocate-session-host-at-logoff/#components-and-workflow)
3. [Setup shutdown and deallocation on disconnect](https://avdpunks.com/2023/01/12/avd-shutdown-and-deallocate-session-host-at-logoff/#setup-shutdown-and-deallocation-on-disconnect)
4. [Conclusion](https://avdpunks.com/2023/01/12/avd-shutdown-and-deallocate-session-host-at-logoff/#Conclusion)

Introduction

Azure Virtual Desktop (AVD) costs come from two sources: underlying Azure resource consumption and licensing.

In the cloud, we want to “Do More, With Less” and maximize the efficiency, ROI and costs of our cloud environment. For AVD this means; the less users are connected to their personal session hosts, the less compute resources should be allocated.

In this article, we’ll explain how to automatically deallocate unused **personal** VMs to increase your cost optimization score.

If you are optimizing pooled or multi session deployments, please take a closer look at the AVD [Scaling Plan](https://learn.microsoft.com/en-us/azure/virtual-desktop/autoscale-scenarios) feature.

Components and workflow

To deallocate virtual machines we use:

* the [Windows Task Scheduler](https://learn.microsoft.com/en-us/windows/win32/taskschd/task-scheduler-start-page)
* an [Azure Function](https://learn.microsoft.com/en-us/azure/azure-functions/functions-overview)

To interrupt the deallocation process we use:

* the Windows Task Scheduler

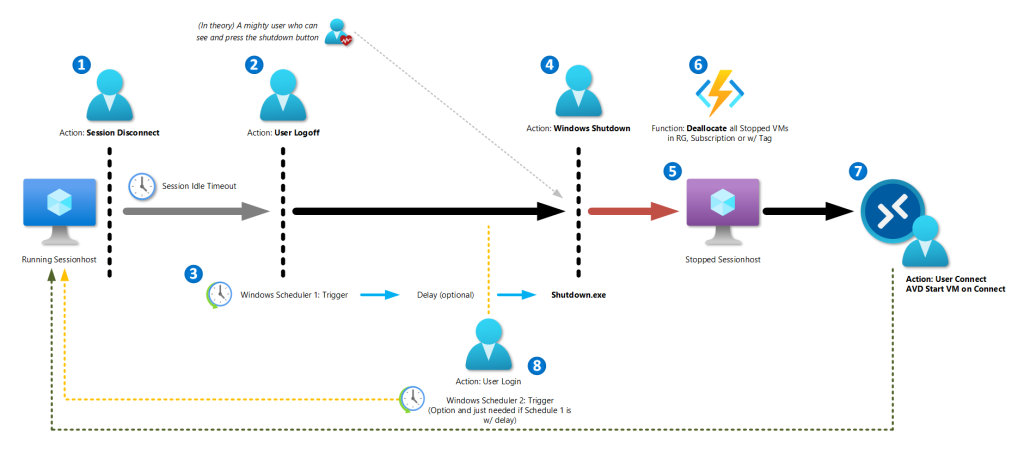
To start the session host again we use:

* the Azure Virtual Desktop [Start VM on Connect](https://learn.microsoft.com/en-us/azure/virtual-desktop/start-virtual-machine-connect) feature.

Lastly, all remote desktop environment settings are done via:

* GPO
* Intune or
* directly via Regkey

This image shows the deallocation workflow:

[](https://avdpunks.com/wp-content/uploads/2024/02/2023-01-06-001-1.png)

1. You can limit the amount of time that active, disconnected, and idle (without user input) sessions remain on the server. The configuration of Timeout and Reconnection Settings for Remote Desktop Services Sessions is documented [here](https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc754272(v=ws.11)).
2. When a user triggers the logoff or the idle time period is reached and the logoff is executed, a **scheduled task** is triggered.
3. The scheduled task is triggers the **Windows Shutdown** and can be **delayed** e.g. for 15min.
4. After defined shutdown delay, the triggered task is initiating the **Windows Shutdown**.
5. As soon as the shutdown is triggered, the session host virtual machines will be in the **Stopped** state
6. An **Azure Function** will deallocate all VMs with the state **Stopped**
7. AVD [Start VM On Connect](https://learn.microsoft.com/en-us/azure/virtual-desktop/start-virtual-machine-connect) will enable the end users to turn on their session host virtual machines when they need them.
8. If a user is triggering a logon while in phase **2.** or **3.** we **terminate** the scheduled shutdown task and wait for the next logoff to be triggered.

Setup shutdown and deallocation on disconnect

By default, AVD session hosts virtual machines (VMs) and Remote Desktop Services allows users to disconnect from a remote session without logging off and ending the session. When a session is in a disconnected state, running programs are kept active even though the user is no longer actively connected. To enable RDP Timeouts for idle session and disconnection you can use one of the following options.

Enable RDP Timeouts

There are several options to enable RDP timeouts. Here you will learn 3 options: via Intune settings catalog, via Active Directoy Group policies and via registry key.

Via Intune Settings Catalog

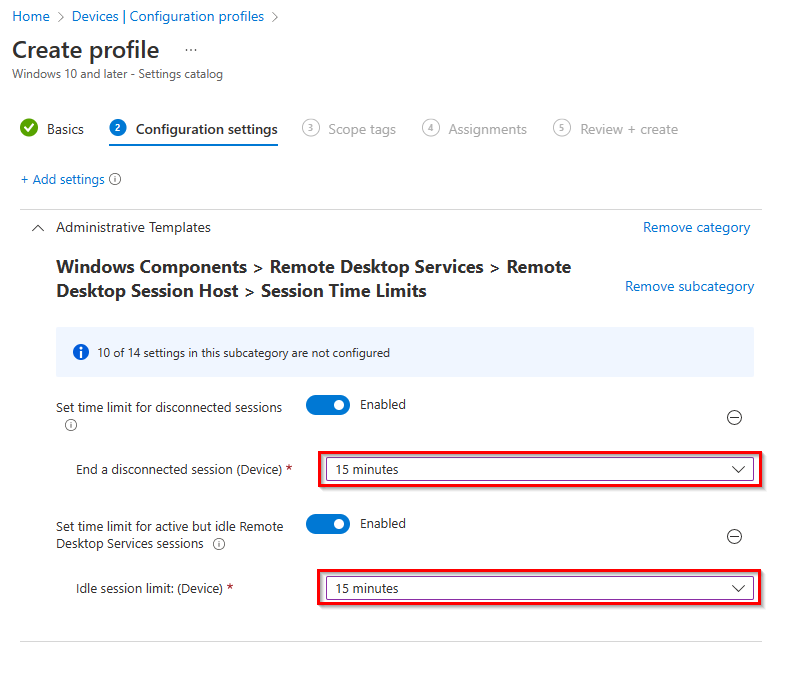
1. First, open the Intune site ([in.cmd.ms](https://in.cmd.ms/)) and then you need to create a new Device Configration profile or use a existing profile to add the RDP timeout settings. The configuration profile can be created in profile type templates or in the settings catalog. The next steps show the RDP settings via settings catalog.
2. Next, search for “**Session Time Limits**” in the settings picker and add the settings “**Set time limit for active but idle Remote Desktop Services sessions**” and “**Set time limit for disconnected sessions**” to your configuration profile.

**Note**: These RDP timeout settings are available as computer and user settings.

[A screenshot of a computer

AI-generated content may be incorrect.](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-10-000.png)

1. Now you need to activate these settings and set your preferred timeouts.

[](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-10-001.png)

1. Then **assign the Intune device configuration to a device or user group**, depending on whether you select the device or user setting.

Via Active Directoy Group policies

You can find the RDP timeout settings under the following path in the GPO editor:

Computer Configuration/Administrative Templates/Windows Components/Remote Desktop Services/Remote Desktop Session Host/Session Time Limits

[A screenshot of a computer

AI-generated content may be incorrect.](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-10-002.png)

Via Registry Key

You can use the following PowerShell commands to add the RDP timeouts by adding two registry keys. The value in milliseconds must be specified in hex format, e.g. 0x000dbba0 = 900000ms = 15mins.

**Note**: You can use the PowerShell function **‘{0:x}’ -f [number in ms]** to convert from decimal to hex.

$registryPath = "HKLM:SoftwarePoliciesMicrosoftWindows NTTerminal Services"

$Name = "MaxDisconnectionTime"

$value = '0x000dbba0'

New-ItemProperty -Path $registryPath -Name $name -Value $value -PropertyType DWORD -Force | Out-Null

$Name = "MaxIdleTime"

$value = '0x000dbba0'

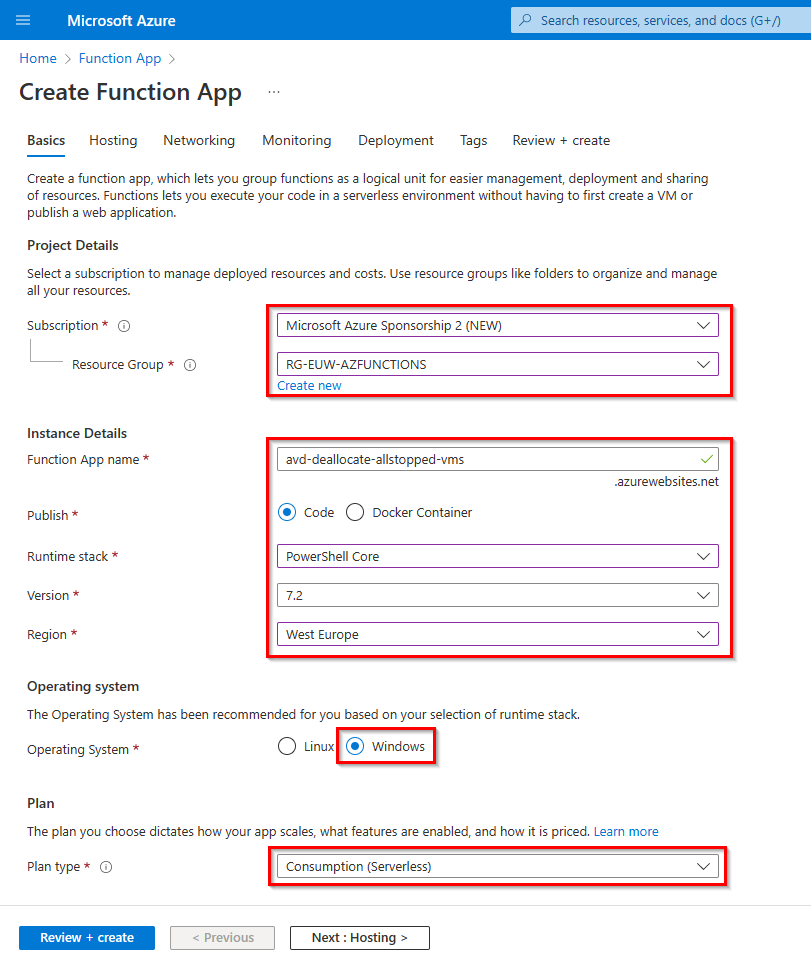
New-ItemProperty -Path $registryPath -Name $name -Value $value -PropertyType DWORD -Force | Out-Null

Create an Azure Function to deallocate all stopped VMs

Azure continues to charge for the VM core hours while it’s **Stopped**. As soon as the VM is deallocated, you just pay for the storage e.g. OS disk and any attached data disks.

So next let’s build a function to deallocate all VMs with the Stopped state.

1. Open the **Azure Function page**, you can use this direct link: <https://azfn.cmd.ms/>, then click **Create** to crate a new Azure function resource.
2. **Select your project details** (subscription/resource group) and then select a **unique Azure function name**, select **PowerShell Core as runtime stack** and **your preferred region**. For the operating system, select **Windows** and the plan type should be **Consumption (Serverless)**.

[](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-002-1.png)

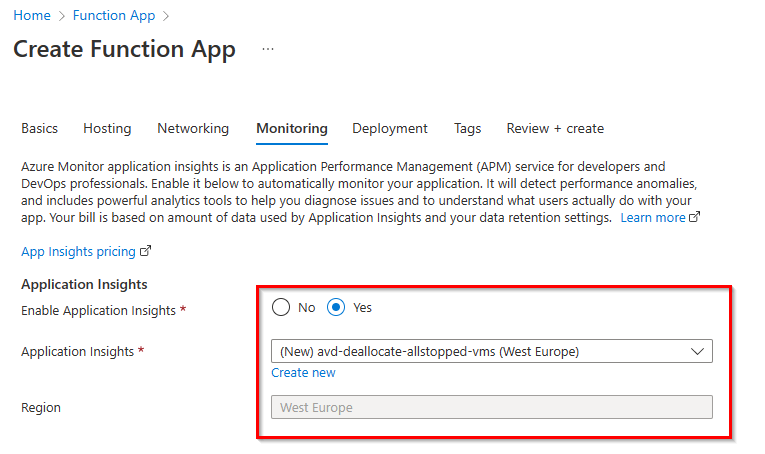
1. Next, on the Hosting tab, **select your preferred storage account or create a new one**.

[A screenshot of a computer

AI-generated content may be incorrect.](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-003-1.png)

1. Next, you can **skip the Network tab** because of the limitation of the service plan. Go to the next tab **Monitoring** and enable it to get more insights into the applications if you want. This is not required, but it will help if a problem occurs.

**Note**: Here you will find some information about [Application Insights](https://learn.microsoft.com/en-us/azure/azure-monitor/app/app-insights-overview?tabs=net).

[](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-004-1.png)

1. The Deploy tab can also be skipped and then you can **assign some tags if you want**. Otherwise, click **Check + Create**.

[A screenshot of a computer

AI-generated content may be incorrect.](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-005-1.png)

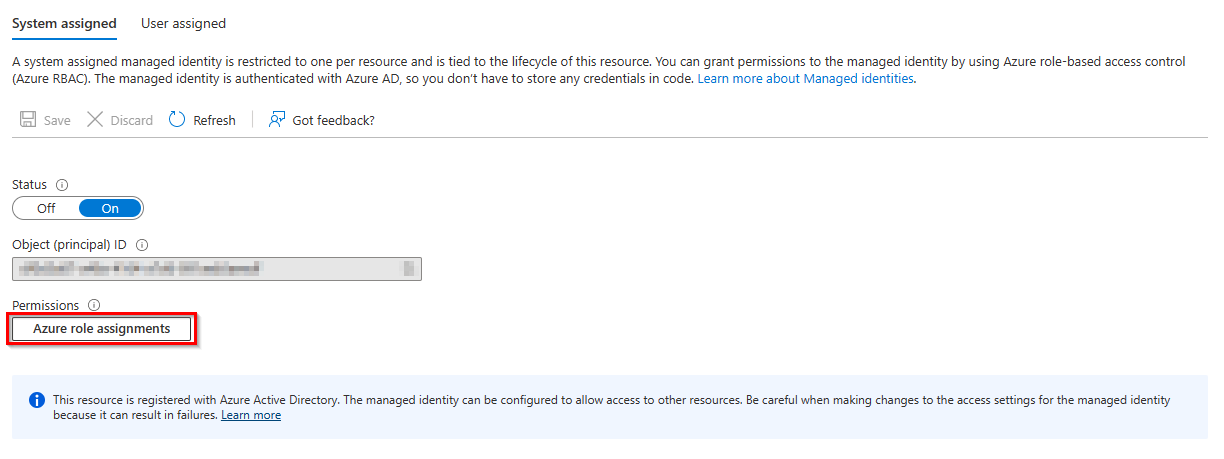
1. When the Azure function is successfully created, access the resource. First, you need to **enable a system-assigned managed identity** and **assign an Azure permission role** that enables virtual machine shutdown. **Open Identity Settings** and change the status for the assigned system from Off to **On**, and then click **Save**.

This registers the Azure function in Azure Active Directoy so that you can assign specific Azure permissions to this managed identity. Alternatively, you can use a user-assigned managed identity instead of a system-assigned one if you wish.

[A text on a white background

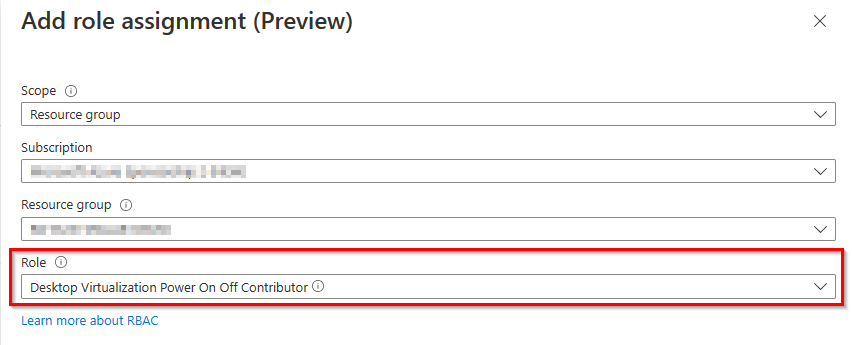
AI-generated content may be incorrect.](https://avdpunks.com/wp-content/uploads/2024/02/2023-01-06-006-1.png)

1. Now you can assign Azure permissions to this managed identity. Click **Azure Role Assignments** and then click **Add Role Assignment**.

[](https://avdpunks.com/wp-content/uploads/2024/02/2023-01-06-007-1.png)

If you want to use this Azure feature only for a specific resource group, select **Resource group as scope** and **your Subscription/Resource group** However, if you want to use it for the entire subscription, select **Subscription as Scope**.

Meanwhile, there is a built-in permission role that allows you to start and deallocate virtual machines. You can use your custom role if you have one, otherwise select the **Desktop Virtualization Power On Off Contributor** role. The same role is also used for the scaling plan feature.

[](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-008-1.png)

1. Next, open the Azure function **Configuration** in the Settings section, as you will need to add a new application setting that defines the resource group.

Click **New Application Setting** and enter **ResourceGroupName** as the application setting name and your value, which should be the resource group where your personal AVD session hosts are stored.

[A screenshot of a computer screen

AI-generated content may be incorrect.](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-009-1.png)

Click **OK** and then click **Save**.

1. For our PowerShell script, some modules are required so that we can log in to Azure and stop some virtual machines. Therefore these modules must be added to the file **requirements.psd1**. For this you have to open **App files** under the functions section and then select **requirements.psd1**.

[A screenshot of a computer

AI-generated content may be incorrect.](https://avdpunks.com/wp-content/uploads/2024/02/2023-01-06-010-1.png)

Add this lines and then click **Save**:

'Az.Accounts' = '2.10.4'

'Az.Compute' = '5.2.0'

[A screenshot of a computer

AI-generated content may be incorrect.](https://avdpunks.com/wp-content/uploads/2024/02/2023-01-06-011-1.png)

1. After you change the app file for the PowerShell modules, you need to restart the entire Azure function. To do this, select **Overview** and click **Restart**.

[A screenshot of a computer

AI-generated content may be incorrect.](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-012-1.png)

1. Now you need to create a function as a timer trigger so that the function is automatically executed every minute or every 5 minutes or whatever. Within your Azure functions, navigate to **Function**, then click **Create** and select **Timer Trigger**. Then enter a descriptive **name for this function** and the **schedule**, e.g. “*/60*” for each minute.

**Note**: The following provides more details on how to define the schedule as NCRONTAB expressions: [NCRONTAB expressions & examples](https://learn.microsoft.com/en-us/azure/azure-functions/functions-bindings-timer?tabs=in-process&pivots=programming-language-python#ncrontab-expressions)

[A screenshot of a computer

AI-generated content may be incorrect.](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-013-1.png)

Click **Create**.

1. Open **Code + Test** in the Developer section and select the PowerShell script **run.ps1**. Then add the following lines to the script and click **Save**.

$StoppedVMs = Get-AzVM -ResourceGroupName $env:ResourceGroupName -Status | Where-Object {($\_.powerstate -eq "VM stopped")}

if ($null -ne $StoppedVMs){

foreach ($VM in $StoppedVMs){

Write-Host "VM $($VM.Name) will be deallocated now..."

$StopVM = Stop-AzVM -Name $VM.Name -ResourceGroupName $env:ResourceGroupName -Force

If ($StopVM.Status -eq "Succeeded") {

Write-Host "VM $($VM.Name) was successfully deallocated..."

} else {

Write-Host ("Something went wrong! Please check the Azure activity log ...")

}

}

} else {

Write-Host ("No VMs could be found in the status stopped...")

}

[A screenshot of a computer

AI-generated content may be incorrect.](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-014-1.png)

**Note**: If you want to change the trigger time, open the **function.json** and change the schedule.

1. Last step, you can try to run a test if everything is configured as excepted, e.g. Azure Permissions for the managed identity. Click **Test/Run** and then click **Run**. You should see all the script output in the logs terminal.

**Warning**: Please run this test in your developer resource group, because if everything is configured as exempt, the Azure feature will deallocate all stopped VMs.

[A screenshot of a computer

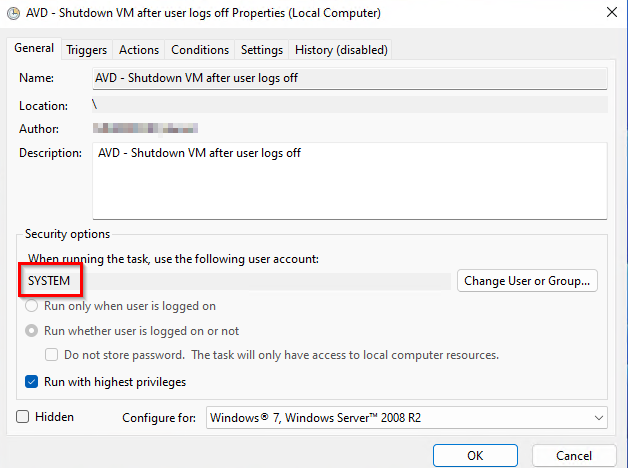
AI-generated content may be incorrect.](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-015-1.png)

Create Windows Task Scheduler for shutdown automation

After you have successfully created an Azure function to deallocate all suspended VMs, you need to create a scheduled task to run on the AVD session host.

First, let’s show what the scheduled task looks like, but you will need to implement the scheduled task as part of your imaging or provisioning process for new AVD session hosts. Below we have added the PowerShell commands to create this task automatically.

The scheduled task must run in a **System Context** and enable **Execute with Highest Privileges**.

[](https://avdpunks.com/wp-content/uploads/2024/02/2023-01-06-016-1.png)

As a trigger, you must select **On Event** and then **Security as Protocol** and **4647 as Event ID**, which means that this task will be taken when this specific Event ID occurs.

**Note**: This task can be started with a delay, e.g. 15 minutes, but this is optional and not required.If you enable the delay, please follow the [optional chapter.](https://avdpunks.com/2023/01/12/avd-shutdown-and-deallocate-session-host-at-logoff/#optional-scheduled-task-to-stop-the-delayed-shutdown-task)

[A screenshot of a computer

AI-generated content may be incorrect.](https://avdpunks.com/wp-content/uploads/2024/02/2023-01-06-017-1.png)

**Event ID 4647** means that the user has initiated a logout, which can be achieved when the user clicks **Sign out** or the RDP connection disconnect timeout reaches its limit and logs the user out.

[A screenshot of a computer

AI-generated content may be incorrect.](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-018-1.png)

Next you need to define the action and select **Start programme**, enter the **path of shutdown.exe** (C:WindowsSystem32shutdown.exe) and add the arguments **/f /s /t 0**.

[A screenshot of a computer program

AI-generated content may be incorrect.](https://avdpunks.azurewebsites.net/wp-content/uploads/2024/02/2023-01-06-019-1.png)

**Warning**: When you start this task, the system shuts down immediately. Before testing, make sure that all your configuration and work is saved.

The following PowerShell script lines create the scheduled task described above.

You can include these script lines for your custom image deployment or in your custom script extension to automatically implement this task for all new AVD session hosts. Otherwise, you can use Intune Script Deployment if you want to enable this task for existing personal VMs.

$TaskName = "AVD - Shutdown VM after user logs off"

$principal = New-ScheduledTaskPrincipal 'NT AuthoritySYSTEM' -RunLevel Highest

$class = cimclass MSFT\_TaskEventTrigger root/Microsoft/Windows/TaskScheduler

$triggerM = $class | New-CimInstance -ClientOnly

$triggerM.Enabled = $true

$triggerM.Subscription='<QueryList><Query Id="0" Path="Security"><Select Path="Security">\*[System[EventID=4647]]</Select></Query></QueryList>'

#A DELAY IS OPTIONAL

#$triggerM.Delay = "PT15M"

$actionM = New-ScheduledTaskAction -Execute "$env:windirSystem32shutdown.exe" -Argument "/f /s /t 0"

$settingsM = New-ScheduledTaskSettingsSet

$taskM = New-ScheduledTask -Action $actionM -Principal $principal -Trigger $triggerM -Settings $settingsM -Description "AVD - Shutdown VM after user logs off"

Register-ScheduledTask $TaskName -InputObject $taskM

**Note**: For troubleshooting, please check the event log of the task scheduler.

(OPTIONAL) Scheduled Task to stop the Delayed Shutdown Task

We all love coffee breaks 😅 you might want to add an delay after the logoff to keep the session host vm up an running to ensure a better user experience.

The delay is configurd and enabled for the scheduled shutdown task. You must also schedule a task to cancel the shutdown task when the user logs back in during to the delay phase.

The following PowerShell commands disable and enable the task **AVD – Shutdown VM after user logs off**:

Disable-ScheduledTask -TaskName 'AVD - Shutdown VM after user logs off'

Enable-ScheduledTask -TaskName 'AVD - Shutdown VM after user logs off'

These PowerShell command lines create the scheduled task to disable and enable the shutdown task at user logon:

Explain

$TaskName = "AVD - Stop the Delayed Shutdown Task"

$principal = New-ScheduledTaskPrincipal 'NT AuthoritySYSTEM' -RunLevel Highest

$triggerM = New-ScheduledTaskTrigger -AtLogOn

$triggerM.Enabled = $true

$actionM = New-ScheduledTaskAction -Execute "$env:windirSystem32WindowsPowerShellv1.0Powershell.exe" -Argument "-command &{Disable-ScheduledTask -TaskName 'AVD - Shutdown VM after user logs off';Enable-ScheduledTask -TaskName 'AVD - Shutdown VM after user logs off'}"

$settingsM = New-ScheduledTaskSettingsSet

$taskM = New-ScheduledTask -Action $actionM -Principal $principal -Trigger $triggerM -Settings $settingsM -Description "AVD - Stop the Delayed Shutdown Task"

Register-ScheduledTask $TaskName -InputObject $taskM

Conclusion

Compute cost of session host VMs are by far the largest cost component. Implementing the process above to automatically turn VMs on at connect and turn them off once the users are no longer connected or they are in idle to long will help you to create a cost-effective AVD environment while avoiding a capital-intensive nightmare.