

Sample Script: Add Devices to Groups Based on User Group Membership Script

1 Summary

This sample script enumerates devices, determines the primary owner and adds the device to a specified group based on the primary owners group membership. The purpose of this is to allow the feature in Intune that applies scope tags to devices to apply scope tags to devices based on user group membership. The current feature in Intune only applies to devices and cannot target user objects.

The current version of the script requires a mapping hash table to be created which maps user group membership to device group membership.

1.1 Script Location

Get the current version of the script from

<https://github.com/scottbreenmsft/scripts/tree/master/Intune/RBAC/DeviceScopeTags>.

1.2 Script Description

This sample script performs the following actions:

1. Authenticates to Azure AD;
2. Get a list of devices to check;
3. For each device returned:
 - a. Get the primary user of the device;
 - b. Get a list of groups the primary user is a member of;
 - c. If one of the groups, the user is a member of is a group in our list of user to device group mappings;
 - i. If the device is not a member of the corresponding device group;
 1. Adds the device to the corresponding device group.

1.3 Script Dependencies

The script requires the Azure AD PowerShell module so that it can use ADAL to authenticate to the Graph API.

1.4 Example script output

```
PS C:\Users\scbree> C:\Users\scbree\OneDrive\Scripts\Intune\ApplyDeviceScopeTag-Runbookcode.ps1
Authenticating...
getting all devices
8 returned.
Processing device: 10001-U-DMPXVP2CJF88. Serial: DMPXVP2CJF88. AADDeviceID= 4dd20d84-2d55-4fd6-a49e-d24dc9087ec. User: tsmith@schools.breenl.com
adding device 77603409-1300-4bf7-872b-d9b9d7884cf2 to device scope tag group af5fa98e-2b94-4cd8-9d3f-ec364882fba5
Processing device: ADMIN. Serial: 8981-6801-1976-3826-3144-9859-47. AADDeviceID= 1a1bf720-95ac-4ae1-a8bd-aef45eb97be. User: breens@schools.breenl.com
Processing device: DESKTOP-UW72VQE. Serial: 2517-5807-8549-0573-4665-4621-86. AADDeviceID= ecd7f1f3-86dd-47dd-b2c2-7a5610188578. User: tsmith@schools.breenl.com
Processing device: S-0728260326-03. Serial: 9119-6690-7485-7438-3607-2826-03. AADDeviceID= 64081931-2256-4145-8e73-a7600b40e28c. User: dem@schools.breenl.com
Processing device: Scott's MacBook. Serial: C02Y8090HH21. AADDeviceID= 1459a67c-6d39-4b9c-afaf-92a9330ff3b6. User: tsmith@schools.breenl.com
Processing device: SHARED-1. Serial: 1425-1545-6990-2562-6349-8345-95. AADDeviceID= cf05c015-7fbc-4c23-b0bb-37189606076b. User: dem@schools.breenl.com
Processing device: STUDENT-1. Serial: 5641-9521-5053-9502-9695-1967-56. AADDeviceID= da5d47fa-c1e8-45cd-9b35-1064330cba05. User: tsmith@schools.breenl.com
Processing device: STUDENT-2. Serial: 2857-1755-8010-6851-0403-2407-59. AADDeviceID= 7afbd1c2-7edd-4595-8fc7-e78b2fec3b0d. User: tsmith@schools.breenl.com
```

1.5 Version History

Version	Date	Description
1	5 th February 2020	Initial Version
2	7 th February 2020	Added detail for running in Azure Automation

2 Setup

2.1 Setup Application Authentication

2.1.1 Create Azure AD Application

The script uses application authentication so it can be scheduled as a task and not require an administrator credentials. The script requires the following permissions:

- Sign in and read user profile
- Read Microsoft Intune devices
- Read and write devices
- Read and write all groups
- Read all users' full profiles
- Read and write all group memberships

Use this article - <https://oofhours.com/2019/11/29/app-based-authentication-with-intune/> - as a guide but add the following permissions:

Device (1)		
<input checked="" type="checkbox"/>	Device.ReadWrite.All Read and write devices ⓘ	Yes
> DeviceManagementApps		
> DeviceManagementConfiguration		
DeviceManagementManagedDevices (1)		
<input type="checkbox"/>	DeviceManagementManagedDevices.PrivilegedOperations.All Perform user-impacting remote actions on Microsoft Intune devices ⓘ	Yes
<input checked="" type="checkbox"/>	DeviceManagementManagedDevices.Read.All Read Microsoft Intune devices ⓘ	Yes
<input type="checkbox"/>	DeviceManagementManagedDevices.ReadWrite.All Read and write Microsoft Intune devices ⓘ	Yes
Group (1)		
<input type="checkbox"/>	Group.Create Create groups ⓘ	Yes
<input type="checkbox"/>	Group.Read.All Read all groups ⓘ	Yes
<input checked="" type="checkbox"/>	Group.ReadWrite.All Read and write all groups ⓘ	Yes
<input type="checkbox"/>	Group.Selected Access selected groups ⓘ	Yes
GroupMember (1)		
<input type="checkbox"/>	GroupMember.Read.All Read all group memberships ⓘ	Yes
<input checked="" type="checkbox"/>	GroupMember.ReadWrite.All Read and write all group memberships ⓘ	Yes

In the end the API Permissions screen should look like this:

Configured 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. [Learn more about permissions and consent](#)

+ Add a permission		Grant admin consent for test_test_Breenlab Academy			
API / Permissions name	Type	Description	Admin Consent Requir...	Status	
▼ Microsoft Graph (6)					
Device.ReadWrite.All	Application	Read and write devices	Yes	✔ Granted for test_test_Br...	***
DeviceManagementManagedDevices	Application	Read Microsoft Intune devices	Yes	✔ Granted for test_test_Br...	***
Group.ReadWrite.All	Application	Read and write all groups	Yes	✔ Granted for test_test_Br...	***
GroupMember.ReadWrite.All	Application	Read and write all group memberships	Yes	✔ Granted for test_test_Br...	***
User.Read	Delegated	Sign in and read user profile	-	✔ Granted for test_test_Br...	***
User.Read.All	Application	Read all users' full profiles	Yes	✔ Granted for test_test_Br...	***

2.1.2 Update script

Update the script with your tenant and app details from the previous step. The tenant is your tenant name which can be found in Azure AD properties, the client ID is the client ID of the app registration you created and the client secret is the client secret key from the app registration you created.

```
#Azure AD App Details for Auth
$tenant = "<tenant>.onmicrosoft.com"
$clientId = "<client ID>"
$clientSecret = "<client secret>"
```

2.2 Configure Groups and Scopes

2.2.1 Groups

For the script to work you'll need a group of users (can include users in subgroups) and a group to add the devices relevant to that user group in.

For example.

User group	Group A Users
Device group	Group A Devices
Scope tag	Group A

2.2.2 Scope tag to group assignment

After you've created the device groups, you need to create/update the scope tag in Intune to tag all devices in the group with the scope tag.

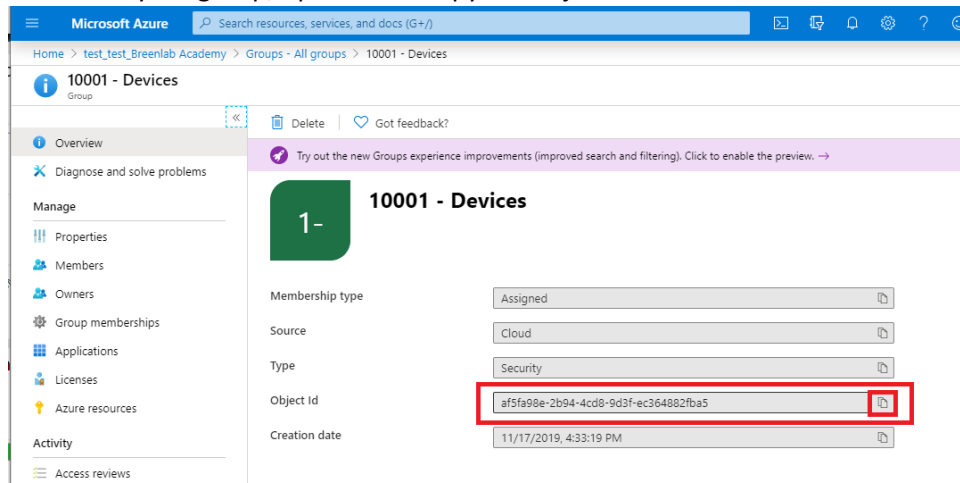
- Navigate to **MEM > Tenant Admin > Roles > Scope tags**;
- Select the scope tag to edit;
- Select groups to assign to;
- Select the relevant device group.

2.2.3 Get Group IDs

We'll need the Azure AD group IDs for the groups in the next step. To get a group ID:

1. Go to the groups blade in Azure AD
(https://portal.azure.com/#blade/Microsoft_AAD_IAM/GroupsManagementMenuBlade/AllGroups)

2. Search for your group, open it and copy the object ID



2.2.4 Update script

There is a hash table at the beginning of the script that maps a user group to the group you want to add the devices to.

For each group you need to duplicate this part:

```
#Record the list of user group to scope tag group mapping here
$UserGroupRoleGroupMapping=@()
$shash = @{}
    UserGroupID      = "66cc746f-1219-4afb-83e1-2fcf96ea4df2" #10001 - Breen Academy North
    ScopeTagGroupID  = "af5fa98e-2b94-4cd8-9d3f-ec364882fba5"
}
$shash = @{}
    UserGroupID      = "0b59cddd-d56c-4857-98d8-f1bb066a947e" #10002 - Breen Academy South
    ScopeTagGroupID  = "23cf75a8-1ba8-4602-bc4a-cdd2d4e39085"
}
$UserGroupRoleGroupMapping+=(New-Object PSObject -Property $shash)
```

So you can see in the example above that all the device that have primary users in the group "66cc746f-1219-4afb-83e1-2fcf96ea4df2" will be added to the device group "af5fa98e-2b94-4cd8-9d3f-ec364882fba5".

3 Script Options

Property	Description
filterByEnrolledWithinMinutes	#change this attribute if you want to get devices enrolled within the last 'n' minutes. #Change this to 0 to get all devices. The time is in minutes. #1440 is 24 hours
useAzureAutomationLastJobTimeAsFilter	[Optional if running from Azure Automation] set the following attribute to true if instead of using "enrolled within last n minutes" #you'd like to control the schedule by getting all devices since the last execution of the #Azure Run book that this script is being run in.
tenant	The Azure AD tenant name (eg. customer.onmicrosoft.com)
clientId	The client ID of the Azure AD App Registration
clientSecret	The client secret of the Azure AD App Registration

runbookName	[Optional if running from Azure Automation] The name of the runbook
rgName	[Optional if running from Azure Automation] The Azure resource group that the Azure Automation account is a part of
aaName	[Optional if running from Azure Automation] The Azure Automation account name
UserGroupRoleGroupMapping	See Configure Groups and Scopes .

4 Run in Azure Automation

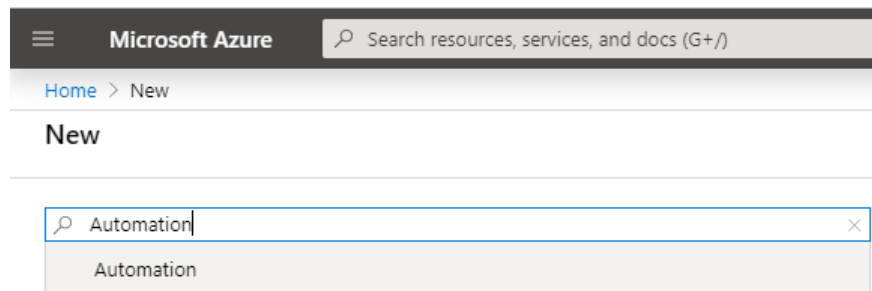
The script can be run from Azure Automation, run manually or scheduled on server. This section provides a guide for setting up the script in Azure Automation.

4.1 Create account

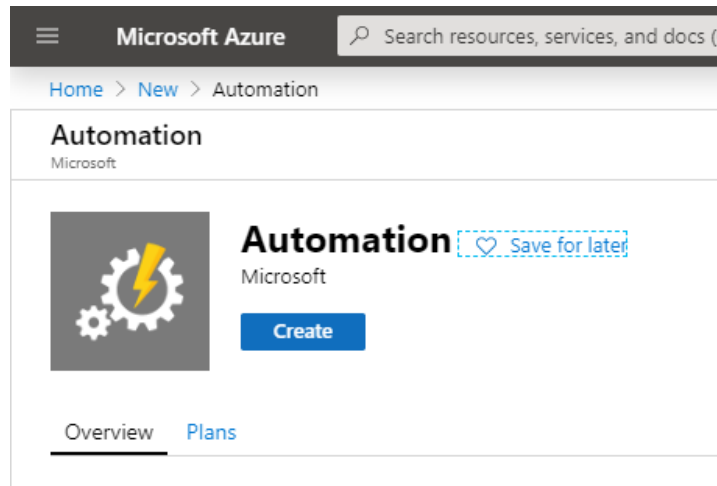
1. Open Azure Portal
2. Click on create a resource



3. Search for Automation



4. Click Create



5. Fill out the details as relevant and ensure you choose to create a Run As account

Microsoft Azure

Home > New > Automation > Add Automation Account

Add Automation Account

Name * ⓘ
IntuneAutomation ✓

Subscription *
Visual Studio Enterprise

Resource group *
(New) IntuneResources
[Create new](#)

Location *
Australia Southeast

Create Azure Run As account * ⓘ
☒ Yes ☐ No

This will create Azure Run As account in the Automation account which are useful for

6. Monitor the creation of the account from notifications. Once created, click on Go to resource

breens@breenlab.scott...
TEST_TEST_BREENLAB

Notifications

[More events in the activity log →](#) [Dismiss all](#) ✓

✓ **New Azure Run As account (service principal) created** ✕
Azure Run As account (service principal) for account 'IntuneAutomation' was created successfully and assigned the Contributor role to this user at the subscription level.
a few seconds ago

✓ **Deployment succeeded** ✕
Deployment 'Microsoft.AutomationAccount' to resource group 'IntuneResources' was successful.
[Go to resource](#) [Pin to dashboard](#)
a few seconds ago

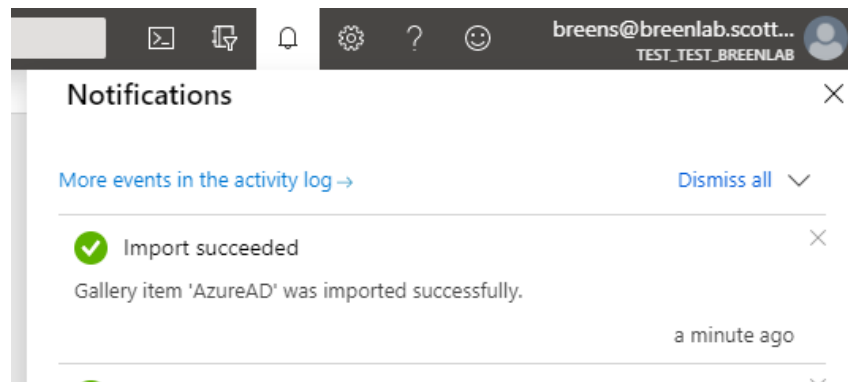
- 7.

4.2 Add module

The Azure AD module needs to be imported into the Automation account.

1. Navigate to the Automation account
2. Under Shared Resources, select Modules Gallery
3. Search for AzureAD
4. Select AzureAD
5. Click Import
6. Click OK

7. Use notifications to monitor the progress



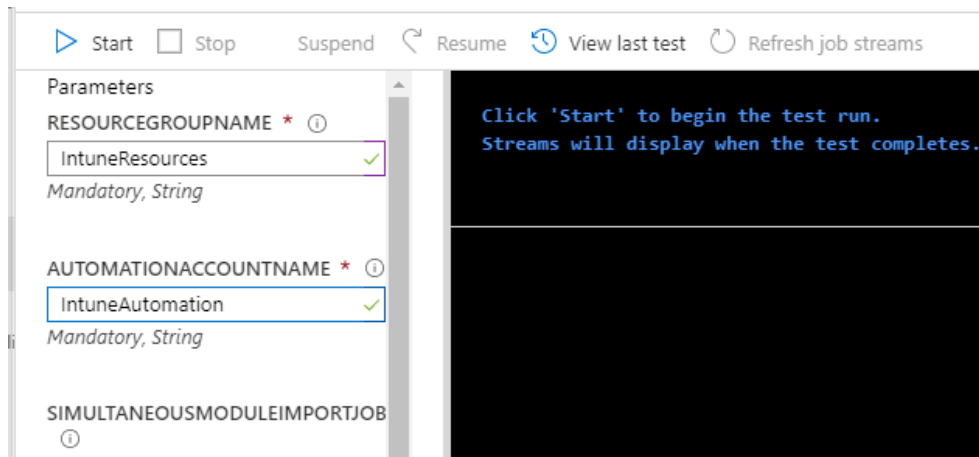
4.3 Update modules

You will need to update the modules in the account using <https://docs.microsoft.com/en-us/azure/automation/automation-update-azure-modules>.

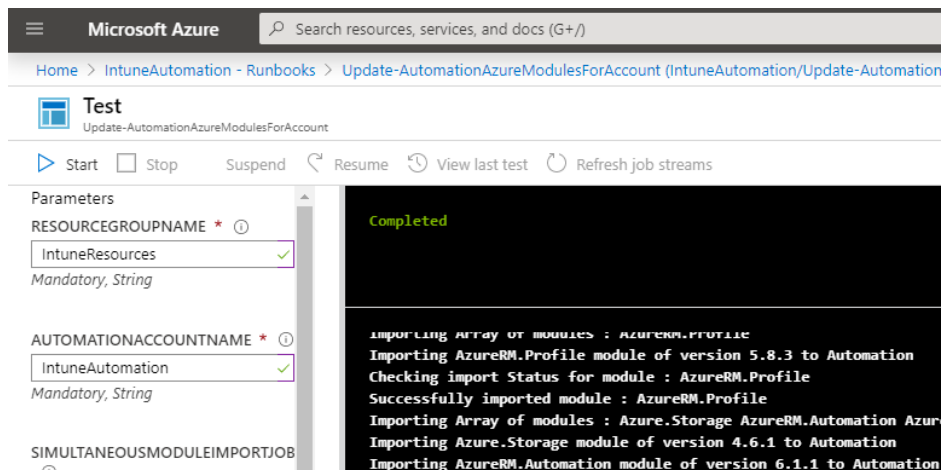
1. Download from <https://github.com/microsoft/AzureAutomation-Account-Modules-Update/releases>
2. Extract the ZIP file
3. Import the runbook Update-AutomationAzureModulesForAccount.ps1

A screenshot of the 'Import a runbook' dialog in the Azure portal. The dialog has a title bar with a tree icon and a close button. It contains four fields: 'Runbook file' with a file icon, 'Name' with a green checkmark, 'Runbook type' with a dropdown menu set to 'PowerShell', and 'Description' with a text area. The 'Runbook file' field contains the text '"Update-AutomationAzureModules...' and the 'Name' field contains 'Update-AutomationAzureModulesForA...'. The 'Runbook type' dropdown is set to 'PowerShell'.

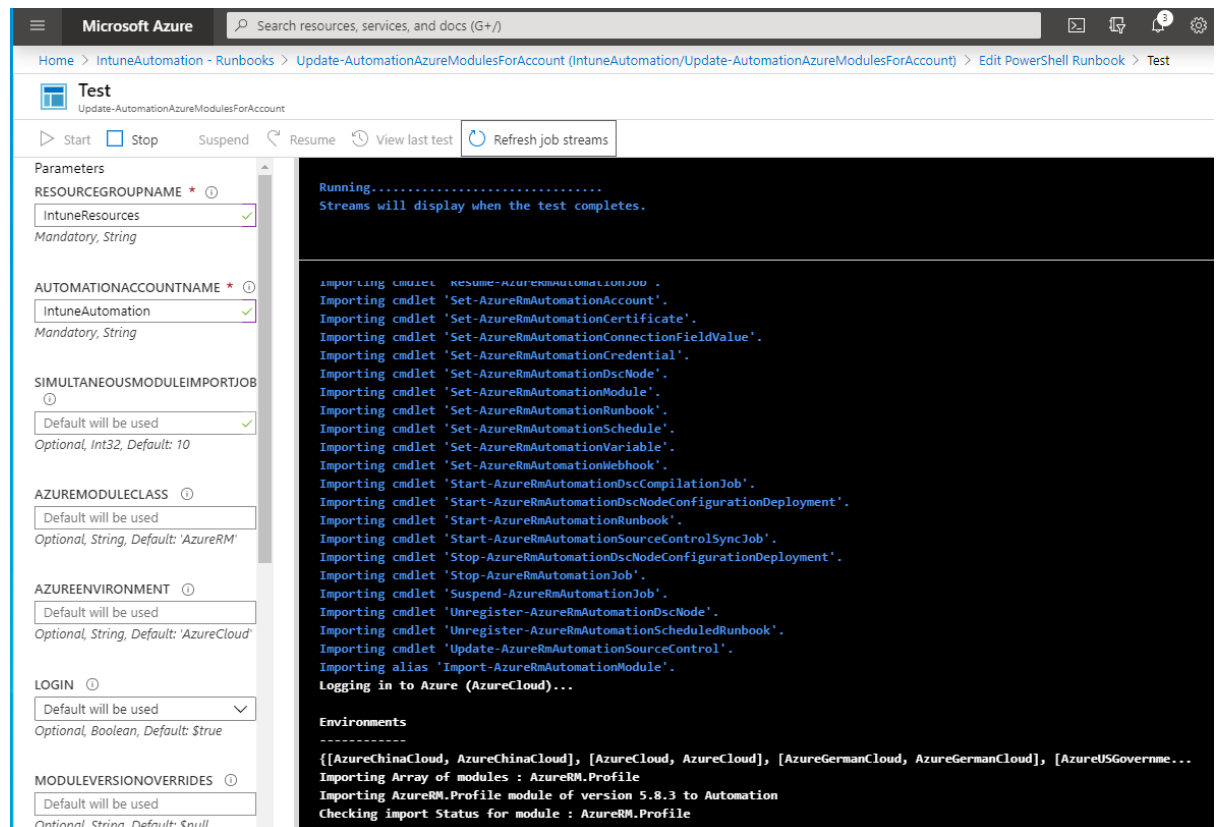
4. Click Test Pane
5. Enter the resource group name and account name from the previous step and click Start



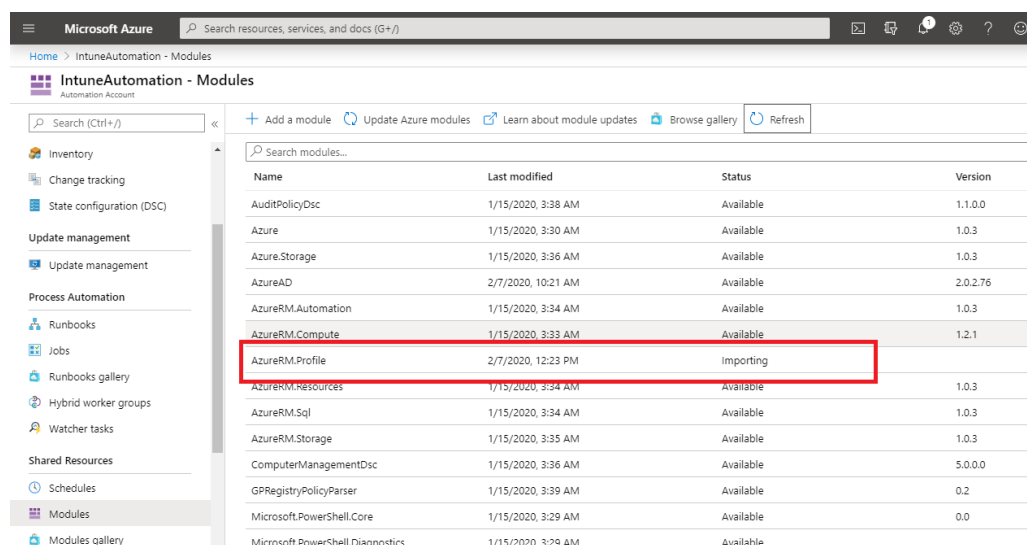
- You can click **refresh job streams** to see status. The screen will update to Completed once done.



Progress:



You can simultaneously monitor the Modules screen on the Automation account and see modules updating:



Name	Last modified	Status	Version
AuditPolicyDsc	1/15/2020, 3:38 AM	Available	1.1.0.0
Azure	1/15/2020, 3:30 AM	Available	1.0.3
Azure.Storage	1/15/2020, 3:36 AM	Available	1.0.3
AzureAD	2/7/2020, 10:21 AM	Available	2.0.2.76
AzureRM.Automation	1/15/2020, 3:34 AM	Available	1.0.3
AzureRM.Compute	1/15/2020, 3:33 AM	Available	1.2.1
AzureRM.Profile	2/7/2020, 12:23 PM	Importing	
AzureRM.Resources	1/15/2020, 3:34 AM	Available	1.0.3
AzureRM.Sql	1/15/2020, 3:34 AM	Available	1.0.3
AzureRM.Storage	1/15/2020, 3:35 AM	Available	1.0.3
ComputerManagementDsc	1/15/2020, 3:36 AM	Available	5.0.0.0
GPRegistryPolicyParser	1/15/2020, 3:39 AM	Available	0.2
Microsoft.PowerShell.Core	1/15/2020, 3:29 AM	Available	0.0
Microsoft.PowerShell.Diagnostics	1/15/2020, 3:29 AM	Available	

4.4 Create Runbook for the script

This section provides instructions for creating the script as a runbook.

1. Navigate to the Automation account
2. Click on Runbooks
3. Click Create a runbook
4. Give the runbook a name

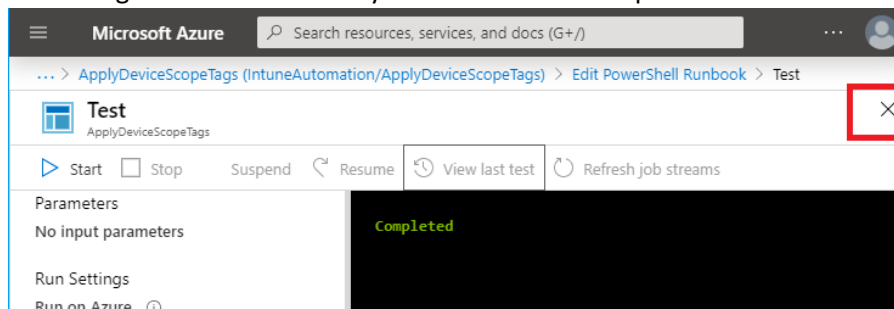
5. Select the type as PowerShell
6. Click Create
7. This will take you to the edit screen
8. Paste the script into the edit screen and then edit the runbook variables

```
#Runbook details if running in Azure Automation to detect last job run time
and to ensure not conflicting with running jobs.
#If the script is being run from an onpremises server or manually, just
leave these attributes as the default, the script will
#ignore them if it isn't being run in the context of Azure Automation.
$runbookName = "<Runbook name>"
$rgName = "<The name of the resource group where the automation account was
created>"
$aasName = "<The name of the automation account you created>"
```

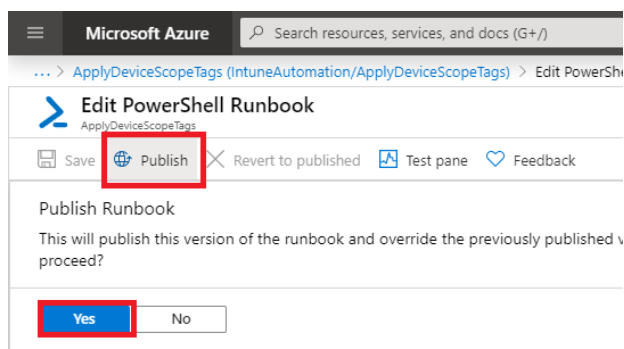
Eg.

```
33 #Runbook details if running in Azure Aut
34 #If the script is being run from an onpr
35 #ignore them if it isn't being run in th
36 $runbookName = "ApplyDeviceScopeTags"
37 $rgName = "IntuneResources"
38 $aasName = "IntuneAutomation"
39
```

9. Save the script
10. Click on test pane
11. Click Start (this will run the script for the first time)
12. Assuming there are no errors you can close the test pane



13. Click Publish, click Yes to confirm the publish.



14. You can now schedule or run the runbook manually. You can view the job status and output from the jobs section.

4.5 Schedule and Script Variables

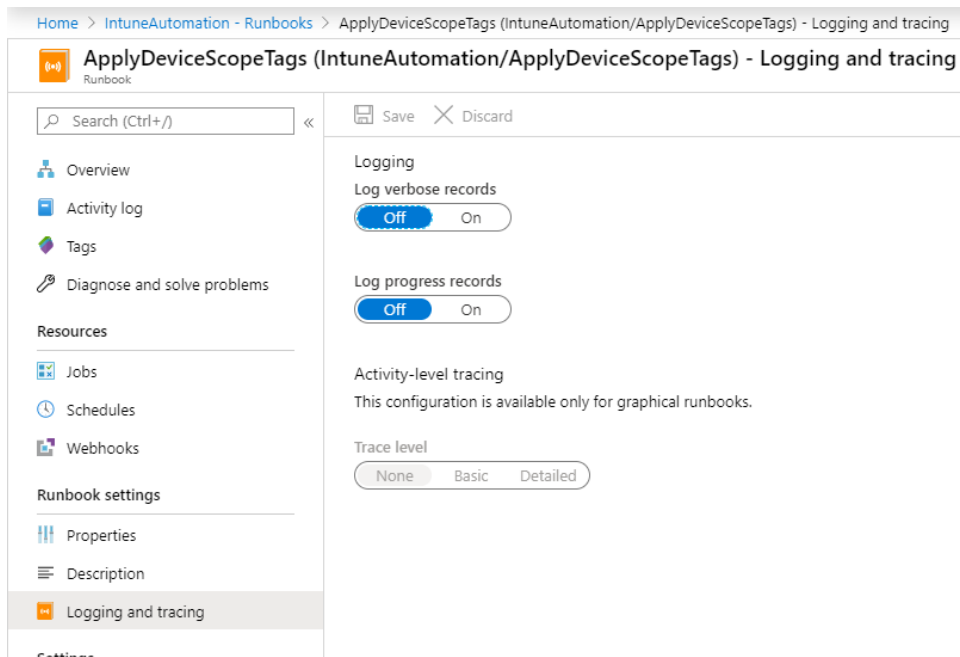
You may want to schedule a script that runs frequently to ensure new enrolled devices are placed into the correct scope tag quickly.

The table below helps determine which Parameters should be set based on the type of job you want to run.

Schedule Type	filterByEnrolledWithinMinutes	useAzureAutomationLastJobTimeAsFilter
Job that runs frequently (maybe every 30 mins) to pick up new devices. Only runs on devices enrolled since the last job run time.	0	\$True
Job that runs once a day or once a week to ensure all devices are categorised correctly regardless of enrolment time.	0	\$False

4.6 Logging Options

The script has some verbose logging which can be enabled in the Runbook from the Logging and tracing section. There are no progress records relevant in this script.



4.7 Example

4.7.1 Example output

DeviceScopeTag 2/4/2020, 10:25 AM

Job

▶ Resume

□ Stop

|| Suspend

🔄 Refresh

Id : e756808a-72a3-49f7-8e93-825478f5a2a5

Status : Completed

Ran ... : Azure

Ran ... : User

Created : 2/4/2020, 10:25:52 AM

Last Update : 2/4/2020, 10:26:57 AM

Runbook : [DeviceScopeTag](#)

Source snapsh... : [View source snapshot](#)

Input **Output** Errors Warnings All Logs Exception

```
Getting last job start time

Authenticating...

getting devices recorded as enrolled since the last runbook execution - 01/13/2020 03:33:11

6 returned.

Processing device: autoenroll_Windows_1/20/2020_10:35 PM. Serial: . AADDeviceID= 00000000-0000-0000-0000-000000000000. User: autoenroll@schools.breenl.com

Processing device: CBeane_Windows_1/20/2020_11:06 AM. Serial: . AADDeviceID= 00000000-0000-0000-0000-000000000000. User: CBeane@schools.breenl.com

Processing device: CBeane_Windows_1/20/2020_9:46 AM. Serial: . AADDeviceID= 00000000-0000-0000-0000-000000000000. User: CBeane@schools.breenl.com

Processing device: DESKTOP-UV72VQE. Serial: 2517-5807-8549-0573-4665-4621-86. AADDeviceID= ec1d71f3-86dd-47dd-b2c2-7a5610188578. User: tsmith@schools.breenl.com

    adding device dc1c90f7-de29-4b25-acd0-914c23960131 to device scope tag group af5fa98e-2b94-4cd8-9d3f-ec364882fba5

Processing device: S-5029695196756. Serial: 5641-9521-5053-9502-9695-1967-56. AADDeviceID= da5d47fa-c1e8-45cd-9b35-1064330cba05. User: tsmith@schools.breenl.com

Processing device: STUDENT-2. Serial: 2857-1755-8010-6851-0403-2407-59. AADDeviceID= 7afbd1c2-7edd-4595-8fc7-e78b2fec3b0d. User: tsmith@schools.breenl.com

    adding device 3ef8b138-f09a-4245-ba89-196393c23566 to device scope tag group af5fa98e-2b94-4cd8-9d3f-ec364882fba5
```

DeviceScopeTag 2/4/2020, 10:25 AM

Job

▶ Resume

□ Stop

|| Suspend

🔄 Refresh

Id : e756808a-72a3-49f7-8e93-825478f5a2a5

Status : Completed

Ran ... : Azure

Ran ... : User

Created : 2/4/2020, 10:25:52 AM

Last Update : 2/4/2020, 10:26:57 AM

Runbook : [DeviceScopeTag](#)

Source snapsh... : [View source snapshot](#)

Input Output **Errors** Warnings All Logs Exception

Errors Warnings

0 0

Type: Any

Search logs...

Time	Type	Details
2/4/2020, 10:26:47 AM	Verbose	Runbook is not already running
2/4/2020, 10:26:48 AM	Output	Getting last job start time
2/4/2020, 10:26:48 AM	Verbose	Last run time was 01/13/2020 03:33:11
2/4/2020, 10:26:48 AM	Output	Authenticating...
2/4/2020, 10:26:49 AM	Output	getting devices recorded as enrolled since the last runbook execution - 01/13/2020 03:33:11
2/4/2020, 10:26:49 AM	Verbose	GET https://graph.microsoft.com/beta/deviceManagement/managedDevices?\$filter=enrolledDateTime ge 2020-01-13T03:33:11Z with 0-byte payload
2/4/2020, 10:26:52 AM	Verbose	received 18884-byte response of content type application/json;odata.metadata=minimal;odata.streaming=true;IEEE754Compatible=false;charset=utf-8
2/4/2020, 10:26:52 AM	Output	6 returned.
2/4/2020, 10:26:52 AM	Output	Processing device: autoenroll_Windows_1/21/2020_10:35 PM. Serial: . AADDeviceID= 00000000-0000-0000-0000-000000000000. User: autoenroll@schools.breenl.com
2/4/2020, 10:26:53 AM	Verbose	POST https://graph.microsoft.com/beta/users/8428b249-afed-49ad-8630-717842b36901/getMemberGroups with -1-byte payload
2/4/2020, 10:26:53 AM	Verbose	received 174-byte response of content type application/json;odata.metadata=minimal;odata.streaming=true;IEEE754Compatible=false;charset=utf-8