**Managing a Microsoft Teams Room (MTR) Device with Intune – Theme**

Centrally managing Microsoft Teams Room devices is a must, especially if you’re in charge of a large fleet of devices, spread across multiple locations.

In this series, we’ll  look at how you can use Microsoft Intune to manage MTR’s. In part 1, we’re focusing on customised themes – the look and feel of the device.

Let’s dive in!

**Bits and Pieces You’ll need**

There’s a few things you’ll need to ensure you have in place first:

* **At least one Windows Based Microsoft Teams Room Device (MTR)**
* **An Office 365 tenant**
* **Suitable licensing for Microsoft Teams and Intune** – more on this below
* **Azure subscription** – You can sign up for $200 free credit to get you started

**Getting things set up**

First things first, I’m starting with an out of the box Microsoft Teams Room (MTR) device. I’ll be using a Logitech TAP, but the actions apply to any windows-based MTR.

**MTR Room Account**

Our MTR will need a room account created for it within Office 365. The exact steps to create your room account will vary depending on if you’re purely within Office 365, or if you’re currently syncing from on premises AD.

Microsoft have the steps to create a room account documented here: <https://docs.microsoft.com/en-us/microsoftteams/rooms/rooms-configure-accounts>

**Licensing the MTR**

There are a number of ways to license an MTR for Intune:

* Meeting Room license
* Microsoft 365 E5 license
* Intune Device license

The simplest (and most cost effective) is to license your MTR with a meeting room license. This license grants the device:

* Teams
* Phone System
* Audio Conferencing
* Microsoft Intune

In this example, I’m going to be licensing my test meeting room:

**testmeetingroom1@chiffers.com**  
  
[Graphical user interface, application

Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/testmeetingroom1.png)

**Licensing an Intune Administrator**

To administer Intune, you’ll also need to grant an Intune license to one or more Intune Service Administrators (or Office 365 Global Tenant Administrators).

Again, there are a number of ways to achieve this. The easiest is to apply a Microsoft 365 E5 license to the administrator, or purchase a separate Intune license. You can even apply a 30 day M365 E5 trial license.

In my example, i’ve created a temporary admin account called **trial-admin@chiffers.com** and have assigned it a trial M365 E5 license.

[](http://blog.chiffers.com/wp-content/uploads/2020/11/intuneadmin.png)

Note that if you don’t license an Intune Administrator and attempt to access the Intune portal (endpoint.microsoft.com), you’ll see various 403 access denied errors.

This account does not need global tenant administrator rights, but does need to be assigned **Intune Admin** rights, under **Manage Roles > Devices**under **Users**in the Office 365 Admin Center.

**Setting up an Azure storage account**

We need to set up a **Storage Account** in Azure to host our XML settings file and customised desktop background image.

To set the storage account up:

1. Sign in to the Azure portal at [**https://portal.azure.com**](https://portal.azure.com/)
2. Search for **Storage Accounts**
3. Click **Create**
4. Fill out the **Project Details**.  
     
   You’ll more than likely need to create a new **Resource Group**, if you don’t already have a suitable one.  
     
   You’ll note too that I’ve changed the replication to **Locally-redundant storage (LRS).** There’s no need to have a geo-redundant storage platform for this use case.  
     
   [Graphical user interface, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/1.-create.png)
5. When ready, click **Review + create**to create the storage account.  
     
   **Note:**Keep in mind that at this point, you’re creating a publicly accessible blob storage account. Don’t put anything on this storage that the general public shouldn’t have access to.
6. Once created, Open **Static Website,** select **Enabled**and click **Save** to turn this feature on.  
     
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/2.-enable-website.png)
7. Make a note of the **primary endpoint URL (this will appear after clicking save)** – we’ll need this later in our **PowerShell script.  
     
   [Graphical user interface, text, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/3.-endpoint.png)**

**Securing Our Storage Account**

Right now, our storage account’s web page (and anything we store on it) is accessible by anyone on the web.  
  
Before we go any further, let’s secure this web site so that only our IP can access it.

1. Click **Networking**
2. Select **Selected Networks** under **Allow access from** and then add the public IP addresses where your MTR’s will access the internet from under **Firewall > Address Range.**You can also tick **Add your client IP address** to add your current public IP to the allowed list. **Tip:**browse to www.whatismyip.com from your MTR device to check its public IP address. **[Graphical user interface, text, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/4.networkfw.png)**
3. When ready, click **Save**

**Configuring our custom theme settings**

To configure various settings on your MTR devices, you’ll need to create an XML file with those settings contained within. You **must** name your XML file **SkypeSettings.xml**

Microsoft have documented all available variables that you can control within the XML file here: <https://docs.microsoft.com/en-us/microsoftteams/rooms/xml-config-file>

In this example, we’re going to be configuring the following settings via the XML file.

* Auto screen share
* Hiding the name of the meeting
* Setting Teams as the default client
* Enabling Bluetooth beaconing
* Disabling auto accept for proximity Join
* Enabling support for SfB meetings
* Enabling support for WebEx meetings
* Enabling custom info for third party meetings
* Disabling dual screen mode
* Where to send log files via email
* Custom Theming

We can safely remove the settings that we don’t want/need from within the file, so long as we keep the formatting the same, as well as the <SkypeSettings> </SkypeSettings> tags at the start and end of the file.

Here’s my XML file: ([**original xml file here**](https://1drv.ms/u/s!AhTKrtxglL1lxRHihldF_hHxcvi8?e=lRemes))

<SkypeSettings>  
<AutoScreenShare>true</AutoScreenShare>  
<HideMeetingName>true</HideMeetingName>  
<IsTeamsDefaultClient>true</IsTeamsDefaultClient>  
<BluetoothAdvertisementEnabled>true</BluetoothAdvertisementEnabled>  
<AutoAcceptProximateMeetingInvitations>false</AutoAcceptProximateMeetingInvitations>  
<SkypeMeetingsEnabled>true</SkypeMeetingsEnabled>  
<TeamsMeetingsEnabled>true</TeamsMeetingsEnabled>  
<WebExMeetingsEnabled>true</WebExMeetingsEnabled>  
<UseCustomInfoForThirdPartyMeetings>true</UseCustomInfoForThirdPartyMeetings>  
<CustomDisplayNameForThirdPartyMeetings>Craig's Conference Room</CustomDisplayNameForThirdPartyMeetings>  
<CustomDisplayEmailForThirdPartyMeetings>testmeetingroom1@chiffers.com</CustomDisplayEmailForThirdPartyMeetings>  
<DualScreenMode>false</DualScreenMode>  
<DuplicateIngestDefault>false</DuplicateIngestDefault>  
<SendLogs>  
<EmailAddressForLogsAndFeedback>mtrfeedback@chiffers.com</EmailAddressForLogsAndFeedback>  
<SendLogsAndFeedback>true</SendLogsAndFeedback>  
</SendLogs>  
<Theming>  
<ThemeName>Custom</ThemeName>  
<CustomThemeImageUrl>mtr-wallpaper.jpg</CustomThemeImageUrl>  
<CustomThemeColor>  
<RedComponent>100</RedComponent>  
<GreenComponent>100</GreenComponent>  
<BlueComponent>100</BlueComponent>  
</CustomThemeColor>  
</Theming>  
</SkypeSettings></SkypeSettings>

**Custom Background Image**

Our custom background image that will be displayed on the room’s displays needs to be exactly **3840×1080 pixels**, and either a JPG, PNG or BMP.

Note that if the room only has a single display, by default the right hand side of the entire image will be displayed on the in-room displays. Dual display rooms will display the full image stretched across both displays.

Microsoft have produced a [custom theme Photoshop template](https://docs.microsoft.com/en-us/microsoftteams/downloads/themingtemplatemicrosoftteamsrooms_v2.1.psd) that can help you design your custom background, ensuring you don’t put graphics over on screen elements like the time.

I’m using a free wallpaper file in this example: **mtr-wallpaper.jpg**

You may wish to add information that users within the room may find useful such as some basic info on using the room, and the helpdesk number to call if there are any issues.

[A bridge over water with buildings in the background

Description automatically generated with low confidence](https://1drv.ms/u/s!AhTKrtxglL1l6Trs3lnkS-uPjluJ)

Now that we have our background image file, and SkypeSettings.XML file we can upload these to our **Azure Storage** account using the **Azure Storage Explorer**

1. Within your storage account in Azure, select **Overview > Open in Explorer  
   [Graphical user interface, application, Word

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/5.explorer.png)**
2. Within the **Storage Explorer app**, expand out your **storage account, blob containers, $web**  
     
   Upload your background image, and SkypeSettings.xml file here  
     
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/6.storage-1.png)

**Deploying the theme via PowerShell**

We need a way of getting our SkypeSettings.xml file, and our custom background image on to each of our MTRs.

To do this, we can use Intune to run PowerShell on each of our MTRs to connect to our storage account, download our SkypeSettings.xml and mtr-wallpaper.jpg file and place them in the correct location on the MTR.

Here’s our script: ([**original file here**](https://1drv.ms/u/s!AhTKrtxglL1l6TmAlPuHDy3O1Fcx))

Invoke-WebRequest -Uri "https://chifferscommtrfiles.z8.web.core.windows.net/mtr-wallpaper.jpg" -OutFile "C:\Users\Skype\AppData\Local\Packages\Microsoft.SkypeRoomSystem\_8wekyb3d8bbwe\LocalState\mtr-wallpaper.jpg"  
Invoke-WebRequest -Uri "https://chifferscommtrfiles.z8.web.core.windows.net/SkypeSettings.xml" -OutFile "C:\Users\Skype\AppData\Local\Packages\Microsoft.SkypeRoomSystem\_8wekyb3d8bbwe\LocalState\SkypeSettings.xml"

you’ll need to edit the locations in red text to point towards your **Azure Storage URL** that you took note of earlier.

Be sure to leave the path to C:\Users\Skype\AppData\Local\Packages\Microsoft.SkypeRoomSystem\_8wekyb3d8bbwe\LocalState\ as it is.

We’ll save this PowerShell script as **mtr-custombg.ps1** on our local machine.

Note: A previous version of this post included the below script. This script no longer works, so please use the script above instead.

**Old Script (no longer works)**

$WebClient = New-Object System.Net.WebClient  
$WebClient.DownloadFile("https://chifferscommtrfiles.z8.web.core.windows.net/mtr-wallpaper.jpg",  
"C:\Users\Skype\AppData\Local\Packages\Microsoft.SkypeRoomSystem\_8wekyb3d8bbwe\LocalState\mtr-wallpaper.jpg")  
$WebClient = New-Object System.Net.WebClient  
$WebClient.DownloadFile("https://chifferscommtrfiles.z8.web.core.windows.net/SkypeSettings.xml",  
"C:\Users\Skype\AppData\Local\Packages\Microsoft.SkypeRoomSystem\_8wekyb3d8bbwe\LocalState\SkypeSettings.xml")

**Upload the script into Intune**

Now we need to upload the script into Intune.

1. Sign in to the Intune portal: [https://manage.microsoft.com](https://endpoint.microsoft.com/)
2. Select **Devices > Scripts**
3. Click **Add > Windows 10 or later  
     
   [Graphical user interface, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/7.-scripts.png)**
4. Give your PowerShell script a name, and click **Next**
5. Upload your**mtr-custombg.ps1** file under **Script Location  
     
   [Graphical user interface, text, application, chat or text message

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/8.-add-script.png)**
6. Click Next
7. On the **Assignments** page select **Selected Groups** and choose the group that contains your **MTR Devices** from the list – for now, our MTR wont be in this list as we haven’t Azure-AD joined in yet. **[Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/assignments.png)**  
     
   This group should contain each of your MTR device computer names.
8. When ready, click **Add** to add your script.

**Note:**Any time you make a change to the script, you MUST change the filename of the script so that Intune knows the file has changed, and to re-run it on the MTR device again.

**Configure Intune Automatic Enrollment**

If you have the correct Azure AD Premium licensing in place, you can use **Intune auto enrollment** to automatically enroll any Azure AD-joined machine to Intune.

If you don’t have the licensing in place, or want to enroll machines manually, you can skip this step.

1. Sign in to the Intune portal: [**https://endpoint.microsoft.com**](https://endpoint.microsoft.com/)
2. Expand **Devices > Enroll Devices > Automatic Enrollment**
3. Set the **MDM User Scope** to **All** (unless you specifically don’t want to add all Azure-AD joined devices to Intune).
4. Leave MAM set to none  
     
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/9.-ae.png)
5. Hit **Save**

Note: If you don’t want all Azure AD joined machines to automatically enroll in Intune, you can select **Some** instead, and then target a security group that contains the user accounts you wish to target for Intune enrollment (ensure this group targets users, not devices).

**Azure AD Join your MTR**

Lets go ahead and Azure AD join our MTR:

1. Log in to the local admin account on the MTR (username: .\admin   password: sfb)
2. Open **Settings > Accounts > Access work or School**
3. Click **Connect**
4. Sign in using your Office 365 meeting room account details (e.g. testmeetingroom1@chiffers.com) – this account MUST be licensed with Microsoft Intune. you could also use an **Intune DEM account** here instead, if you wish.  
     
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/10.-sign-in-azure-ad.png)
5. After a few moments, your MTR will be joined to Azure Active Directory. If you have enabled automatic enrollment for all devices, your MTR will be automatically enrolled in Intune too.

Now we need to add our MTR to our **MTR Devices** group in Azure AD:

1. Open the Azure Active Directory Portal > Groups > MTR Devices (or what ever you’ve named your group)
2. Add our newly Azure-AD joined device into this group.

**Two Extra Steps if you don’t have automatic enrollment enabled in Intune:**

1. Open **Settings > Accounts > Access work or School**
2. In the sign in screen that appears, enter your meeting room account details again and sign in (e.g. testmeetingroom1@chiffers.com)

Once you have enrolled the device in Intune, you’ll need to wait a while for the device to connect to the Intune service and download the **Microsoft Intune Management Extension.**

This extension will then automatically run the PowerShell script, pulling down the SkypeSettings.XML and mtr-wallpaper.jpg file to the MTR.

**Test!**

**Now that everything is in place, let’s test to make sure everything  is working as expected**

1. Log in to your MTR as a **local admin.**
2. Open **Windows Explorer**, and browse to c:\program files (x86)
3. Check to ensure that the **Microsoft Intune Management Extension folder**exists  
     
   [[Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/IntuneManagement.png)\](http://blog.chiffers.com/wp-content/uploads/2020/11/IntuneManagement.png)
4. Next, browse to **C:\Users\Skype\AppData\Local\Packages\Microsoft.SkypeRoomSystem\_8wekyb3d8bbwe\LocalState** and confirm that**mtrbg.png** and **SkypeSettings.XML** exist within the folder  
     
   [Table

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/files.png)
5. Restart the MTR and confirm that the new background image is displayed on the in room displays

**Making Changes to the Theme**

From time to time, you’ll more probably want to update the theme and push those changes out to your MTRs.

To ensure your changes appear on your MTR devices, it’s important to understand how often Intune runs PowerShell scripts.

The **Intune management extension agent** checks with Intune once every hour and after every reboot for any new scripts or changes.  Once the script executes, it doesn’t execute again unless there’s a change in the script or policy. If the script fails, the Intune management extension agent will attempt to retry the script three times for the next 3 consecutive Intune management extension agent check-ins.

This means that if you wish to publish a new MTR background image, you’ll need to upload the image to your **Azure Storage Account**, and then re-upload your **PowerShell** script into Intune – even if your script hasn’t had any changes made to it.

Be sure to name your new MTR background image differently to your current image too – I’ve noticed that if the old and new image names are the same, they don’t update.

If you’d like to read up further on PowerShell scripts and Intune, Microsoft have documented the process here: <https://docs.microsoft.com/en-us/mem/intune/apps/intune-management-extension>

**Troubleshooting**

One of the most common issues that can occur is that the **Intune management extension agent**does not install on the MTR.

To solve this problem, first ensure the device is both Azure-AD joined, and enrolled correctly in Intune.

Next, ensure you’ve waited enough time (at least an hour) for the Intune management extension agent to download and install on the device.

Another common issue is that the **theme doesn’t apply.**Be sure that your **SkypeSettings.XML** file contains the correct file name and extension for your wallpaper file, and that your PowerShell script contains the correct URL to download the file from your Azure storage account. Check too that you can browse to the storage account from the MTR.

**Regedit**

Downloaded intune policies are stored within the Registry in the following location:

**Computer\HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\IntuneManagementExtension\Policies\**

[Graphical user interface, text, application

Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2020/11/11.-regedit.png)

Expanding the keys under **Policies**, you can drill down and see the current status (ResultDetails) of your PowerShell script. In the above example, we can see that the script experienced an issue downloading files.

The **DownloadCount** shows the number of times the script attempted to run (3 in the above example).

A script will run once per check-in period. If the script successfully runs, it will never run again. If the script fails, it will retry up to 3 times before failing and stopping forever.  
  
We can force a re-run of the above script by updating the **DownloadCount** to **0**, and setting **Result** to blank and then restarting the **Intune Management Extension** service. Note though that this will only re-run the currently downloaded script. If you’ve made changes to your PowerShell Script (and renamed it!), you’ll need to wait for the next Intune check-in for the new script to be downloaded to the machine and then ran.

**Final Thoughts**

Managing your MTR deployment via Intune is straightforward, and enables you to have greater control of the end user experience.

**Managing a Microsoft Teams Room (MTR) Device with Intune – Updates**

**Updates – a hot topic**

Love them or hate them, updates are an integral part of keeping your IT environment up to date, secure and running smoothly. Microsoft Teams Room devices run a version of Windows 10, and Microsoft strongly recommends that you keep both the OS, and MTR app patched and up to date.

Of course, you may already be aware that the MTR does a fine job of managing its own updates today via a built in scheduled task that runs every night at 2am. The task runs a PowerShell script that queries Windows Update and the Microsoft Store for Business to obtain the latest **supported** OS updates and Microsoft Teams Room system app, before installing them and automatically rebooting the MTR where the device signs back in as the local Skype user, launches the MTR app and signs itself back into Teams ready for another day of Teams meetings.

**Hang on – What are Supported OS Updates?**

You’ll notice I underlined and bolded **supported** OS updates in the above paragraph. Microsoft rigorously tests the MTR app before confirming that a particular version of Windows is given the tick of approval to confirm that its supported.

It’s important to check the [**Microsoft Teams Rooms app version support**](https://docs.microsoft.com/en-us/microsoftteams/rooms/rooms-lifecycle-support) to confirm which specific versions of Windows 10 are listed as supported, before attempting to push a version to your MTRs via Intune. Installing a more up to date version of Windows that hasn’t been tested yet may mean you experience undesired results on the MTR.

**Why would I want to control my updates via Intune?**

Great question! In general, I always recommend that my customers utilise the MTRs built in update patch management where possible.

However, there are scenarios where customers may wish to manage these updates via Intune instead. Common scenarios include:

* Wanting to manage the version of Windows installed across all devices within the org
* Wanting to ensure the device meets a certain compliance threshold
* Wanting to manage updates using another form of update distribution – such as WSUS
* Wanting control over when updates are installed

**Windows Updates vs Teams App Updates**

One important thing to note is that while you can control Windows updates on the device, the MTR app updates that come from the Windows Store cannot be controlled via Intune.

It is strongly recommended that you allow the MTR to download updates from the Windows Store for Business itself. If however you block access to the Windows Store for Business, you can manually deploy MTR updates by downloading the latest MTR app, and deploying it manually via PowerShell. I’ve documented this approach here: [**https://blog.chiffers.com/2021/09/06/manually-update-an-mtr-with-the-latest-teams-app/**](https://blog.chiffers.com/2021/09/06/manually-update-an-mtr-with-the-latest-teams-app/)

**Configuring an Update Policy for an MTR**

Let’s go ahead and setup our update policy in Intune to manage updates on our MTR.

1. Login to the Intune portal at **manage.microsoft.com**
2. Expand **Devices > Windows > Update rings for Windows 10 and Later > Create Profile  
   [Graphical user interface, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/1.-Create-profile.png)**
3. We’ll give our profile a name: **MTR\_Updates  
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/2.name_.png)**
4. Next we’ll configure our update ring for our MTR devices.

|  |  |  |
| --- | --- | --- |
| **Item** | **Setting** | **Notes** |
| Servicing Channel | Semi-Annual Channel | Don’t choose Windows Insider. |
| Microsoft Product Updates | Allow |  |
| Windows Drivers | Allow |  |
| Quality update deferral period (days) | 0 | The number of days to defer quality (security updates and bug fixes etc) updates before installing them |
| Feature update deferral period (days) | 0 | The number of days to defer feature (new versions of Windows 10) updates before installing them |
| Set feature update uninstall period | 60 | The number of days you can uninstall a feature update on an MTR before the files are removed |
| Automatic update behavior | Auto install and restart at maintenance time |  |
| Active hours start and end | 3AM til 7AM | I’d recommend a 4 hour window |
| Restart checks | Skip |  |
| Option to pause Windows Updates | Disable |  |
| Option to check for Windows Updates | Disable |  |
| Require user approval to dismiss restart notification | No |  |
| Remind user prior to required auto-restart with dismissible reminder (hours) | blank |  |
| Remind user prior to required auto-restart with permanent reminder (minutes) | blank |  |
| Change notification update level | turn off all notifications, including restart warnings | You may choose to leave these enabled if you wish |
| Use deadline settings | Not configured |  |

1. [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/3.updatering-1.png)
2. Lastly, assign the policy to your group of MTR devices.

Once that’s done, we’ll need to create **Feature**and **Quality** update rules.

1. Click **Feature updates for Windows 10 and later** on the left, then click **Create profile  
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/5.-Feature-updates.png)**
2. Give it a name: **MTR Feature Updates**and be sure to select either **Windows 10, version 20H2**, or **Windows 10, version 1909  
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/6.-Feature-updates-select.png)**
3. Add your MTR device group to the list of **Included Groups**
4. Click **Create** to create the Feature Update policy

**Now we’ll create our Quality Update** policy

1. Choose **Quality Updates for Windows 10 and later > Create Profile**
2. Give the policy a name: **MTR Quality Updates**
3. Be sure to select an up to date version of quality updates
4. Select the number of days to wait before a restart is enforced. I’d recommend 1 day, but keep in mind that this may enforce the MTR to restart during business hours.  
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/7.-Quality-updates.png)
5. Assign the policy to your group of MTR devices, and click **Create**.

That’s it! Your MTRs are now configured to have their updates managed by Intune. Your MTR app updates will still come from the Windows Store for Business (unless you’ve specifically blocked this from occurring).

**Managing a Microsoft Teams Room (MTR) Device with Intune – Configuration Profiles**

**What are configuration profiles?**

Configuration profiles are very similar to Group Policies – they let you control many different aspects of the operating system like blocking access to USB storage devices, or setting a custom background image.

**Should I create configuration profiles for my MTRs?**

Unless you have a specific requirement, my recommendation is to avoid creating overly restrictive configuration policies for your MTRs.

MTRs are essentially meeting room appliances, not a typical Windows 10 machine that an end user uses. The end user has no access to anything outside the MTR app, including no access to removable storage, programs, a web browser or files stored on the MTR or in the cloud.

Don’t apply your end user compute targeted Windows 10 configuration policy to your MTRs – you’ll break things.

**What are the recommended profiles to configure for an MTR?**

Typical use cases for configuration profiles include enabling **Bitlocker** to encrypt the local drive.

**Add a new Configuration Profile**

Let’s add a new configuration profile for our MTRs

1. Select **Device > Configuration Profiles > Create Profile**
2. From the blade that appears, choose **Windows 10 or Later** under platform, and select **Templates** under profile type  
   [Graphical user interface, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/1.-Create-profile-1.png)
3. Next, select the Template name you wish to configure.  
     
   Based on this post [Managing Teams Meeting Rooms with Intune – Microsoft Tech Community](https://techcommunity.microsoft.com/t5/intune-customer-success/managing-teams-meeting-rooms-with-intune/ba-p/1069230) from the Intune Support team, Microsoft recommend the following:

|  |  |
| --- | --- |
| **Template Name** | **Can you configure this template for MTR use?** |
| Administrative Templates | Yes – Although I would not recommend this. |
| Certificates | Yes – if required within your environment. |
| Delivery Optimization | Yes – If you’d like to configure this within your environment. |
| Device Firmware Configuration Interface | Check for supported hardware [here](https://docs.microsoft.com/intune/configuration/device-firmware-configuration-interface-windows) |
| Device restrictions | Yes – Although I would not recommend this. |
| Edition Upgrade | Not supported |
| Email | Not recommended |
| Endpoint Protection | Yes – You can enable bitlocker encryption. |
| eSim | Not supported |
| Identity Protection | Not supported |
| Kiosk | Not supported |
| Powershell Scripts | Yes (Devices need to be Azure AD or Hybrid Azure AD Joined |
| Shared multi-user device | Not supported |
| VPN | Not recommended |
| Wi-Fi | Not recommended (Some devices to have Wifi built in to them, but its always best to cable them into the network via Ethernet. |
| Windows Information Protection | Not recommended |

1. Once you have selected the template you wish to configure give the template a meaningful name
2. Configure the profile
3. Assign the profile to your MTR device group and click create.

**My Environment Recommendations**

In my test environment, I have Bitlocker enabled, as well as remote desktop and remote PowerShell (although these two aren’t necessary as you shouldn’t ever need to RDP to the MTR itself (admin profile only), and you can run PowerShell scripts via Intune).

Here’s how I enabled these settings:

**Enabling Bitlocker**

Most people like to ensure Bitlocker is enabled on their devices within their environment.

If you’re wanting to enable Bitlocker on your MTR devices, check first that **your MTR supports TPM 2.0** (you can check this by running**tpm.msc** on the device under admin mode) and that UEFI boot is enabled in the BIOS.

To enable Bitlocker on the MTR, we’ll create a configuration profile:

1. Sign in to the Intune portal at **https://manage.microsoft.com**
2. Under **Devices > Configuration Profiles**click **Create Profile**
3. Select **Windows 10 and Later** and **Templates** then choose **Endpoint Protection** from the list and click **Create  
   [Application

   Description automatically generated with medium confidence](http://blog.chiffers.com/wp-content/uploads/2021/09/2.-new-compliance-policy-bitlocker.png)**
4. We’ll give our new profile a name  
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/3.-enable-bitlocker.png)
5. Next, under **Configuration Settings > Windows Encryption**ensure the following is enabled/set. You dont need to enable/set anything further to enable Bitlocker base config  
   [Graphical user interface

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/4.-select-bitlocker.png)
6. We’ll assign our policy to our **MTR Devices** group  
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/5.-groups.png)
7. We can skip past assigning any **Applicability Rules** as our group contains our MTR devices.
8. Lastly click **Create** to create the policy

**Enabling Remote Desktop and Remote Powershell**

Although not required, you may wish to enable remote desktop to allow you to connect to the MTR’s admin login via RDP from your internal network. You may also wish to enable remote Powershell if you want to remotely run commands on the MTR – although generally speaking this isnt recommended as you can run PowerShell commands on the MTR from the Intune portal.

However, here’s how to enable these features through Intune if you wish to:

1. Let’s start by creating a new Configuration Profile  
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/6.-rdp-profile.png)
2. Under **Configuration Settings** select **All Settings** on the left, then search for “Remote Desktop”
3. Select the Settings Name **Allow Users to connect remotely by using Remote Desktop Services  
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/7.-rdp.png)**
4. In the pop out that appears, select **Enabled** and click **OK  
   [Graphical user interface, text, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/8.-rdp-enabled.png)**
5. Next, search for **WinRM**and select **Allow remote server management through WinRM  
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/9.-winrm.png)**
6. In the pop out that appears, select **Enable** and then either enter \* in IPv4 and IPv6 filter dialog boxes, or specify IP addresses/ranges that you wish to allow to connect to the MTR via remote PowerShell (these MUST be within your environment, and be Azure AD/Hybrid Azure AD joined).  
   [Graphical user interface, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/9.-winrm-enabled.png)
7. Click **OK** to save the changes.
8. Click **Next**, and skip assigning **Scope Tags**.
9. On the **Assignments** page, be sure to select your**MTR Devices**group and click Next
10. Click **Create** to create and save the profile

**Now we need to go and create firewall rules to allow connectivity to our MTR**

1. Select **Endpoint Security** on the left, then select **Firewall  
   [Graphical user interface, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/10.-fw.png)**
2. Click **Create Policy** to create a new policy
3. Select **Windows 10 and Later** and **Microsoft Defender Firewall Rules** and click **Create**  
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/10.-fw-create.png)
4. We’ll give our firewall profile a name  
   [Graphical user interface, text, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/11.-fw-new.png)
5. Under **Microsoft Defender Firewall**, click the arrow on the right to expand the list, and then click **Add  
   [Graphical user interface, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/12.-fw-add.png)**
6. We’ll create our Allow RDP rule first:  
   [Graphical user interface, text, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/13.fw-rdp.png)  
   Note that i’ve specifically only allowed subnet 192.168.1.0/24 to access my MTR’s over RDP.
7. Sit **Save**to save the rule.

**Now let’s create the Remote Powershell rule**

1. Click **Add**, and give the rule a name and add the following details:  
   [Graphical user interface, text, application, chat or text message

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/14.fw-rps-1.png)
2. Once done click **Save**
3. Be sure to assign the rule to your **MTR Devices** group.
4. Then click **Review + Create**to save the firewall rules.

**Lastly (You’re almost there!) we need to run a Powershell script on our MTR’s to set the local network connection as private**

1. Create a **.ps1** file with the following Powershell command:  
     
   **Set-NetConnectionProfile -NetworkCategory Private -Verbose**
2. In the Intune portal, go to **Devices > Scripts** and click **Add**
3. Give the script a name  
   [Graphical user interface, text, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/15.-set-nw-profile.png)
4. Under **Script Settings** upload your PS1 file, and leave everything else as default and click **Next**
5. Assign this to your **MTR Devices** group and click next, then **Create**to save the script.  
   [Graphical user interface, text, application, chat or text message

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/16.-enable.png)
6. Now you need to wait an hour for the MTR to do its hourly check-in with Intune, download the scripts and apply them!

**Updating the local admin password**

A common ask is to update the local admin password on each MTR to something other than the default “sfb” password.

This is easily done via Powershell

1. Create a .PS1 file with the following lines:  
     
   **Set-localuser -name Admin -Password (ConvertTo-SecureString -AsPlainText “sfb123!” -Force) -AccountNeverExpires**
2. In the Intune portal, go to **Devices > Scripts** and click **Add New**
3. Give your script a name:  
   [Graphical user interface, text, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/17.-LA-pass.png)
4. On the **Script Settings** page, add your PS1 file, and be sure to select **Yes** next to **Run script in 64 bit Powershell Host** and click **Next  
   [Graphical user interface, text, application, chat or text message

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/18.-upload-script.png)**
5. **Assign the script**to your **MTR Devices** group, and then review and create it.

**Managing a Microsoft Teams Room (MTR) Device with Intune – Autopilot**

**What is Autopilot?**

Windows Autopilot is a way of automating the OOBE (Out of Box Experience) when setting up a new Windows device. Without autopilot, you’d typically need to enter the language, keyboard input, user account details, Azure AD Join, Networking and Wi-Fi details and more. Autopilot can automate all these steps so that all you need to do is power on the device, and allow it to sign in.

**Is Autopilot supported today with MTRs?**

In short, no Autopilot is not officially supported by Microsoft for MTRs yet. However, based on my testing, autopilot does work for MTRs.

**What can I configure on an MTR with Autopilot?**

You can automate the following (and more!):

* Language and keyboard settings
* Azure AD Join
* Device naming (including automatically naming the device based on its serial number, or a random string of numbers)
* Networking and Wi-Fi details (no authenticated proxies though – MTRs do not support authenticated proxy servers).
* Any setup PowerShell scripts/configuration profiles including device wallpaper, Bitlocker requirements etc.

**What doesn’t currently work with Autopilot on an MTR?**

Today, you can’t configure the login username and password for the room inside the Teams room app on the MTR. This would still need to be manually entered by the MTR installer.

**Configuring Autopilot Profile**

**Let’s go ahead and set Autopilot up for our MTR.**

1. Login to the Intune portal at **https://manage.microsoft.com**
2. Go to **Devices > Windows > Windows Enrollment > Deployment Profiles**
3. Click **Create Profile > Windows PC  
   [Graphical user interface, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/1.-create-ap-profile.png)**
4. Give your profile a name, and be sure to select **Yes** under **Convert all targeted devices to Autopilot  
   [Graphical user interface, text, application, email

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/2.-name.png)**
5. Under **Out of box experience (OOBE)**, select the following:  
   [Graphical user interface, text, application

   Description automatically generated](http://blog.chiffers.com/wp-content/uploads/2021/09/3.-oobe-1.png)  
   Under **Enter a name,**you can use the %SERIAL% macro to add the device serial number to the computer name, or %RAND:5% to add 5 random numbers to the computer name. In my example, i’ve added MTR-%SERIAL% meaning my MTR device will be named MTR-119872130 (if the serial number of my device is 119872130)
6. Be sure to assign the profile to your **MTR Devices**group, and click **Create** to create the profile.

**Add MTR devices to Autopilot List**

Now that our profile is configured, let’s add our MTR to our list of Windows autopilot devices

We’re going to need to grab the device hash of our MTR device to import it into the Azure portal.

**Note:** Some manufacturers will supply you with a CSV file containing the device hashes that you can import into the Azure portal.

There’s two ways to do this.

**If you already have an MTR up and running:**Run the following in PowerShell on the MTR

1. New-Item -Type Directory -Path “C:\HWID”
2. Set-Location -Path “C:\HWID”
3. Set-ExecutionPolicy -Scope Process -ExecutionPolicy RemoteSigned
4. Install-Script -Name Get-WindowsAutoPilotInfo
5. Get-WindowsAutoPilotInfo -OutputFile **AutoPilotHWID.csv**

**If you have a new MTR fresh out the box:**  
Power on the MTR and launch the OOBE. Press Shift+F10 to open the Powershell window over the top of the OOBE, then run the following:

1. Set-ExecutionPolicy bypass
2. Install-Script Get-WindowsAutoPilotInfo
3. (be sure to select Y for each prompt that appears).
4. Get-WindowsAutoPoilotInfo -online

**Once you have ran one of the above two options:**

1. Go to **Devices > Windows > Windows Enrollment > Devices**If you used the first method that generates the **AutoPilotHWID.csv**file, you can import this file using the **Import** button.  
     
   If you used the fresh MTR out of the box method, you’ll see the MTR’s serial number listed (normally within 10 or so minutes).

**The Autopilot Experience on the MTR**

Once you have configured Autopilot and added your devices to the list, ensure your MTR is connected to the network via an Ethernet cable (This won’t work over Wi-Fi!), boot your MTR and allow it to go through the Out of Box Experience (OOBE).

[A computer screen with a blue background

Description automatically generated with low confidence](http://blog.chiffers.com/wp-content/uploads/2021/09/MTR-ap-scaled.jpg)

Your MTR should automatically boot into the local Skype profile and prompt you for the Office 365 room account details. Depending on what configuration profiles and PowerShell scripts you’ve assigned to your MTR devices group, your device should also have applied custom wallpaper, an updated local admin password, Bitlocker enabled, Wi-fi details etc. It will also be automatically joined to Azure Active Directory and be renamed using your Autopilot device name settings.

**Final Thoughts**

And there you have it! A way of simplifying the out of box experience for your MTR devices to make the installation even simpler.

Feel free to leave your comments/questions below.