Managing Microsoft Teams in Windows- Guide for Admins

**Intro\Background:**

**Microsoft Teams is so much more than just a chat service. It provides us a unique platform to perform a variety of tasks apart from sending messages like- sharing of files, calling, checking feeds etc. Moreover, as Microsoft Teams tightly integration with Office 365, we don’t need to switch apps when we collaborate on a document. And as Teams is a fully cloud-based solution, the move to the cloud is seamless. We have a wide range of 3rd party apps that integrate into Teams so that our users get the most seamless experience.**

**In this article we are going to discuss the various methodologies by which Microsoft Teams can be managed from an admin perspective for a Windows device.**

Chart, radar chart

Description automatically generated**Different ways of managing Microsoft Teams as an Admin**

**As seen above, there are many ways of administering Microsoft Teams as an Admin. Each approach provides us different levels of control and we can always deploy a combination of the above as per the requirement to best fit the needed usecase**. **Now we will take a look over each of these approaches individually.**

**Managing Microsoft Teams via Conditional Access policy:**

**Conditional access is a feature of Azure which allows\restricts users from accessing any cloud hosted application after the authentication has been done, by subjecting the request to specific conditions. We can leverage CA policy in our organization and impose the appropriate conditions to the user’s request while they are trying to access Microsoft Teams.**

**There are a variety of conditions that we can leverage like, allowing access only-**

* **When the request is coming from a specific IP range**
* **When the request is coming from a Intune enrolled and Compliant device**
* **Prompt for a MFA**
* **When the request is coming from a device which is Hybrid Azure AD Joined**
* **A combination of all the above settings**

**Policy Settings needed:**

**Below is the policy setting that needs to be configured by the admin in order to implement Conditional access on Microsoft Teams**

Graphical user interface, application

Description automatically generated**Conditional Access policy in the Azure Portal**

**As seen above, we have created a Conditional access policy, targeted it to Cloud Resource(Microsoft Teams), selected the relevant OS(Windows), selected the relevant client apps(browser and desktop clients) and imposed the needed condition(requiring a Intune compliant device.**

**End user experience:**

**The user gets the below message while trying to log into Microsoft teams via the application or the browser and is not authorized to access Microsoft teams as his device is not enrolled and compliant to Intune**

Graphical user interface, text

Description automatically generated with medium confidence**Accessing Teams via the Desktop App in Windows 10**Graphical user interface, application, Word

Description automatically generated**Accessing teams via Browser**

**As an admin we can track this request from the user’s sign in logs in the portal as well as by referencing the corelation id** **as shown below-**

* **The below screenshot from the portal shows the name of the CA policy which blocked the user from accessing Microsoft Teams**

Graphical user interface, text, application, email

Description automatically generated

* **The below screenshot from the portal gives more details to the admin and helps him understand why the access to Teams was blocked for the user and more relevant details**

Graphical user interface, text, application, email

Description automatically generated**Sign-in logs for the user from the Azure portal**

**Once the device is enrolled, the user is authorized to access the Teams application as shown below**

Graphical user interface, text, application, Word, email

Description automatically generated**Device is now enrolled and compliant to Intune**Graphical user interface, application

Description automatically generated**User is now able to access Microsoft Teams**

**Conclusion\My 2 cents:**

* **Using Conditional access requires Azure AD P1 license which is a part of EMS E3\E5**
* **Conditional access is a very subtle way of putting in another layer after authentication before the user will have access to the Teams application**
* **Using conditional access we can allow\restrict user’s access to the Teams app. Once the user is inside the application, CA has no control over the session\user’s experience**

**Managing Microsoft Teams via Windows Information Protection (WIP) policy:**

**Windows Information Protection (Formerly known as enterprise data protection) is a feature built into Windows 10 that allows IT shops to control and manage business data separately from personal data on users' devices. We can impose WIP policy with the appropriate settings to the Teams application running on user's device**

**Policy setting needed**

* **As seen below, we can make a WIP policy from Intune and target it to the Microsoft Teams Desktop application**

Graphical user interface, text, application, email

Description automatically generated**WIP policy setting at the portal- applying the same to Microsoft Teams Desktop application**

* **In the below setting we will select “Block” which would restrict and movement of data between the Teams application and any other application in the device.**

Graphical user interface, text, application

Description automatically generated**WIP Policy setting at the Intune portal restricting movement of data to Personal applications**

**End user Experience**

* **When the end user launches the Teams application, he can see the name of his domain as well as a brief case icon, signifying that the Teams app is now a WIP managed application.**

Graphical user interface, application

Description automatically generated**Teams application running in Enterprise context at the user’s device**

* **As seen in the task manager, Microsoft Teams is now running in Enterprise context as it is managed, and not in Personal context like the other applications**

Table

Description automatically generated**Task manager from the user’s device**

* **Now if the user tries to leak any data from Teams, by copying anything from the chat and pasting it to any application, the action is prohibited**

Graphical user interface, application, Teams

Description automatically generated**Data leakage stopped via WIP in Teams application at user’s end**

**Conclusion\My 2 cents:**

* **Using WIP policy is a very good way of implementing something like a DLP on Teams application**
* **This ensures that the coporate data in Teams application stays within the coporate boundaries and is not leaked to personal applications like- notepad or other browsers**
* **WIP policy can be applied to a device from Intune, irrespective of whether the device is enrolled and managed via Intune or not**

**Managing Microsoft Teams via ADMX policy:**

**We have ADMX available, which is pre-imported in Microsoft Intune. We can leverage the ADMX and deploy device configuration policies to Microsoft Teams and administer the same**

**Policy Setting Needed:**

* **We can make the ADMX policy for Microsoft Teams which is a device configuration policy in Intune**
* **The device needs to be enrolled to Intune, in order to implement this policy**

Graphical user interface, text, application, email

Description automatically generated**ADMX policy setting at the Intune portal**

* **In the below policy setting we can define the tenant id of our organization**
* **By this setting, only the organization’s account can we used to log into the Microsoft Teams application and personal account login will be blocked**

Graphical user interface, text, application, email

Description automatically generated**Policy at the Intune portal whitelisting the organization’s tenant id**

**End user Experience**

* **As seen below, when the user tries to log into the Microsoft Teams desktop application via a personal account or any account which does not belong to the tenant mentioned in the policy by the admin- the login is blocked**

Graphical user interface, text, application, Teams

Description automatically generated**User blocked while logging into Microsoft Teams with a personal account**

* **When a user from the tenant allowed by the admin in the policy tries to log into the Teams application- he is able to do the same**

Graphical user interface, application

Description automatically generated**User allowed to log into Teams application while using an account from organization’s tenant**

**SideNote-** **The same settings (as in ADMX) are available in ‘Policy for Office Apps’ section in the Intune portal as shown below**

Graphical user interface, application, Teams

Description automatically generated

**Conclusion\My 2 Cents:**

* **Using ADMX and restricting only tenant’s account to log into the Teams application is a very viable approach specially if the device is COD**
* **This ensures that application is strictly being used for corporate purpose.**
* **The control available in ADMX via Intune is a limited with only few controls available at this point.**

**Managing Microsoft Teams by targeting the Registry directly**

**Every setting that we are deploying via any channel(MDM or other) is eventually targeting a registry hive at the device. We always have the option of targeting the registry directly and make the needed changes. Below is an example wherein we are regulating the registry via a powershell script which would prohibit the launch and usage of Microsoft Teams application in the device**

**Settings needed:**

* **One of the ways of automating the creation of registry in devices is via powershell.**
* **We can use a small script like below which would create a Reg key to disallow the run of teams.exe**

A picture containing text

Description automatically generated

**Now we can run this script manually on the device or deploy it via any delivery mechanism like SCCM\Intune or 3rd party softwares like Bigfix etc.**

Graphical user interface, text, application, email

Description automatically generated**Powershell script run at the user’s machine**

**End user experience**:

* **At the user’s end, once the scripts run, we see the relevant keys have been created in the registry**

Graphical user interface, application

Description automatically generated**Registry changes with respect to the powershell script**

* **User can no longer launch and run the Teams desktop application** **via any means(directly\going to the exe and launching\powershell etc)**

Graphical user interface, text, application, email

Description automatically generated**Launch of Teams application denied at user’s dektop**

**Conclusion\My 2 Cents:**

* **Targeting the registry is also one of the ways of administering the Teams application, if we are aware of the exact registry path that is to be targeted**.
* **I would personally recommend following this approach only for settings which are not available via other means (like MDM\Admin Center) to avoid complications**.

**Managing Microsoft Teams via Applocker**

**Managing Microsoft Teams in a windows device can also be done via Applocker. However this approach is limited to allowing\blocking the launch of the Teams app in the device**

**Settings Needed:**

* **We can create an Applocker executable rule via the local policy editor**
* **In this rule we can deny the run of Teams application as shown below**

Graphical user interface, text, application

Description automatically generated**Applocker created at the desktop – denying the launch of Teams application**

* **We can launch the created Applocker file in a browser and it looks like the below**

Graphical user interface, text, application, email

Description automatically generated**Applocker XML blocking Teams app**

**Conclusion\My 2 Cents:**

* **Using applocker is also a commonly used method to restrict the launch of the Teams application**
* **This method is also limited to allying\denying the launch of the app and we cannot control any other setting related to Teams via this methodology**

**Managing Microsoft Teams via Teams Admin Center**

**The admin center delivers a customizable and tailored experience and provides us actionable intelligence**

**From the admin center we can deploy org wide settings\user specific settings which determines their in-app experience**

**There are lots of customizations available in form of policies in the Teams Admin Center**

**Policy Settings needed:**

* **Depending upon the kind of setting needed, we will have to set the appropriate policy at the admin center.**
* **In the below example we will block the chat for users\Prohibit screen sharing\Create a Teams channel\Remove application from the app center\prohibit external chatting by the members..**

Graphical user interface, application

Description automatically generated**Policy Setting at the Teams Admin Center**Graphical user interface, application

Description automatically generated**End user behavior- Chat option is missing**

Graphical user interface, text, application

Description automatically generated**Admin creating a Teams channel at the Teams Admin Center**Graphical user interface, application, website

Description automatically generated

Graphical user interface, text, application

Description automatically generated**Admin disabling screen sharing and chat for the users**

Graphical user interface, application

Description automatically generated**End user experience- Screen sharing and chat is disabled**

Graphical user interface, application, Teams

Description automatically generated**Admin blocking Applications from the Teams Admin Center**

Graphical user interface, application

Description automatically generated**Blocked apps not visible to the user in App store**

**Conclusion\My 2 Cents:**

* **Managing Microsoft teams via the Teams admin center is the most preferable approach**
* **It provides us a variety of, and a very granular in-app control**
* **Using this approach we can determine how the app experience of different users as per the org requirement**

**Managing Microsoft Teams via O365 Admin Center:**

**There are some settings in O365 admin center which are applicable on Teams as well. This approach provides us some additional controls which are inherently not there in the Teams Admin Center**

**Policy Settings Needed:**

* **We can create a DLP policy in the O365 admin center**
* **We have a variety of controls here like- Prohibiting users from using obscene language, sharing confidential credit card details etc**

Graphical user interface, text, application

Description automatically generated**DLP policy created by the admin in O365 admin center**Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated**Blocking the sharing of sensitive information in the policy**

**Conclusion\My 2 Cents:**

* **O365 admin center provides us many additional settings like Supervision\DLP which are very useful**
* **If used in conjunction with the policies from Teams admin center, most of the use cases are met for the organization**

**Managing Microsoft Teams- Deployment via MDM(Intune)**

**Intune being a delivery mechanism allows us to deploy Microsoft Teams app to the devices. We can deploy Teams along with the Microsoft suite or as a standalone application**

**Policy Setting needed:**

**Below is the deployment that needs to be made by the Intune admin in order to push the Microsoft Teams app to users**

**This technique is only applicable to devices enrolled to Microsoft Intune**

Graphical user interface, application

Description automatically generated

**End User Experience:**

* **The end user can now see the Microsoft Teams app deployed to his device and present in the control panel as seen below**

Graphical user interface, application

Description automatically generated

* **In the eventlogs at the machine we can see the relevant successful installation events.**

Table

Description automatically generated**Eventlogs from the device suggesting successful installation of Microsoft Teams**

**Conclusion**:

* **As seen above** **in this post, there are various ways of administering Microsoft Teams via different methodologies**.
* **Each approach provides us different level of control and helps us manage different settings**
* **Each approach is not a replacement of the other, they are to be used in conjunction so a to get the most granular control on the application as per the organization’s requirement**!