

## Setup guide Monitoring script

1. Create user which will send notification emails and assign Exchange online P1 license to this user, call this user for example Account Monitoring
  2. Create or copy the recipient address(es) , they need at least to have Exchange online P1 license as well
  3. Create app in Entra ID  
Entra.microsoft.com -> Applications -> App registration -> New registration
- 1) Give app a name for example MonitoringSignInTool
  - 2) Choose Account in this organizational directory only

Home > Users > BG > App registrations >  
**Register an application** ...

\* Name  
The user-facing display name for this application (this can be changed later).  
 ✓

Supported account types  
Who can use this application or access this API?  
 Accounts in this organizational directory only (Need 4 Cloud only - Single tenant)  
 Accounts in any organizational directory (Any Microsoft Entra ID tenant - Multitenant)  
 Accounts in any organizational directory (Any Microsoft Entra ID tenant - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)  
 Personal Microsoft accounts only

[Help me choose...](#)

Redirect URI (optional)  
We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Register an app you're working on here. Integrate gallery apps and other apps from outside your organization by adding from [Enterprise applications](#).

By proceeding, you agree to the [Microsoft Platform Policies](#) 

**Register**

4. Now go to API Permissions -> Add a permission -> Microsoft Graph -> Application permissions -> Search for AuditLog.Read.All -> Select and choose Add permissions

The screenshot shows the Azure portal interface for managing API permissions. On the left, the 'MonitoringSignInTool | API permissions' blade is open, showing the 'Configured permissions' section. It lists 'User.Read' under 'Microsoft Graph (1)' with 'Delegated' type and 'Sign in and read user profile' description. A note indicates that admin consent is required. On the right, a 'Request API permissions' dialog is displayed for 'Microsoft Graph'. The 'AuditLog (1)' section is expanded, showing 'AuditLog.Read.All' selected with 'Yes' for 'Admin consent required'. Other options like 'AuditLogsQuery-CRM' and 'AuditLogsQuery-Endpoint' are listed but not selected. At the bottom of the dialog are 'Add permissions' and 'Discard' buttons.

5. Now search for mail and select Mail.Send and press Add permissions

This screenshot shows the same process as the previous one, but for the 'Mail' API instead of Microsoft Graph. The 'MonitoringSignInTool | API permissions' blade shows 'AuditLog.Read.All' and 'User.Read' under 'Microsoft Graph (2)'. On the right, the 'Request API permissions' dialog is shown for the 'Mail (1)' section. 'Mail.Send' is selected with 'Yes' for 'Admin consent required'. Other options like 'TeamsActivity' and 'VirtualAppointmentNotification' are listed but not selected. The 'Add permissions' button is visible at the bottom of the dialog.

## 6. Press Grant admin consent for “Your org”

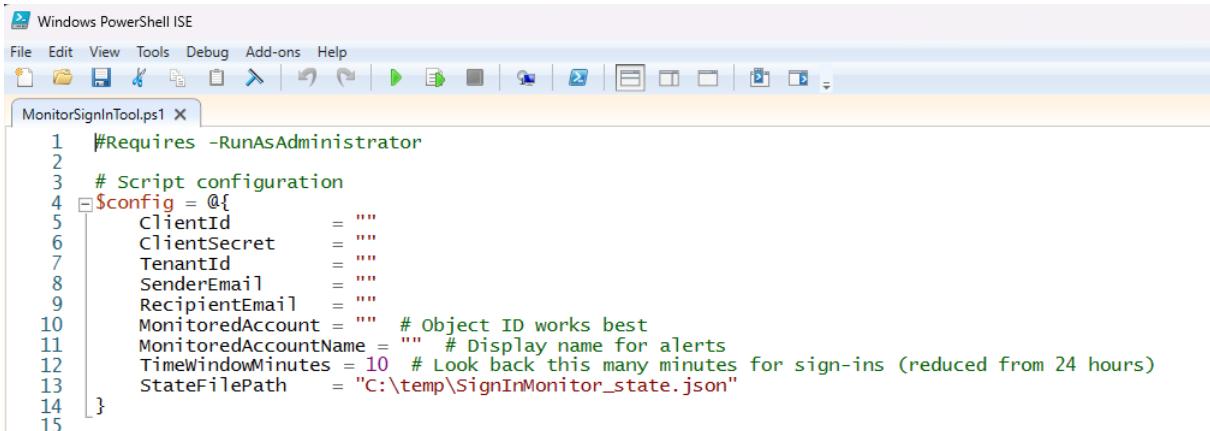
The screenshot shows the Azure portal's API permissions page for the application 'MonitoringSignInTool'. The left sidebar includes options like Overview, Quickstart, Integration assistant, Diagnose and solve problems, Manage (with Branding & properties, Authentication, Certificates & secrets, Token configuration, API permissions), Expose an API, App roles, Owners, Roles and administrators, Manifest, Support + Troubleshooting, and New support request. The main content area displays 'Configured permissions' for Microsoft Graph, listing three permissions: AuditLog.Read.All (Application, Read all audit log data, Yes, Not granted for Need 4...), Mail.Send (Application, Send mail as any user, Yes, Not granted for Need 4...), and User.Read (Delegated, Sign in and read user profile, No). At the top, there are two warning messages: one about editing permissions and another about tenant-wide consent revoking previous grants. A prominent red box surrounds the 'Grant admin consent for Need 4 Cloud' button.

7. Confirm and check that Status says Granted on the right pane
8. Now go to Certificates & secrets and press New Client Secret
9. Give it a description, choose number of days and press Add

The screenshot shows the Azure portal's Certificates & secrets page for the application 'MonitoringSignInTool'. The left sidebar is identical to the previous screenshot. The main content area shows a table for client secrets, which is currently empty. To the right, a modal dialog titled 'Add a client secret' is open, containing fields for 'Description' (set to 'MonitoringSignInToolClientSecret') and 'Expires' (set to 'Recommended: 180 days (6 months)').

10. Copy the Value (not secret ID) and save it for later
11. Now go to Overview -> and copy Application (client) ID and Directory (tenant) ID and save it for later

12. Find downloaded MonitorSignInTool.ps1, right click and choose edit



The screenshot shows the Windows PowerShell ISE interface with the file 'MonitorSignInTool.ps1' open. The code is a PowerShell script with the following content:

```
1 #Requires -RunAsAdministrator
2
3 # Script configuration
4 $config = @{
5     ClientId      = "..."
6     ClientSecret   = "..."
7     TenantId       = "..."
8     SenderEmail    = "..."
9     RecipientEmail = "..."
10    MonitoredAccount = "" # Object ID works best
11    MonitoredAccountName = "" # Display name for alerts
12    TimeWindowMinutes = 10 # Look back this many minutes for sign-ins (reduced from 24 hours)
13    StateFilePath   = "C:\temp\SignInMonitor_state.json"
14 }
15
```

Insert your values between columns

- 1) ClientId is your Application (client) ID
- 2) ClientSecret is your Value ID from Certificate & Secrets
- 3) TenantId is Directory (tenant) ID
- 4) SenderEmail is your sender email address
- 5) RecipientEmail is your recipient email address
- 6) MonitoredAccount is Object ID of account which you want to monitor ( you can find in Entra -> Users. PS: It's best to use ObjectId instead of UPN to avoid big delays)
- 7) MonitoredAccountName is the address or name of account which is monitored (this is optional if you don't want to get notified with objectId)
- 8) TimeWindowMinutes is the scope for how many minutes back it should check sign in logs for the specific user. You can edit the value to TimeWindowHours if you prefer to use hours instead. Don't forget to edit file path if your script is located somewhere else and not in C:\temp

The script remembers last check time and saves ID's in SignInMonitor\_state.json file which will be generated at first time you run the script, to avoid duplicate checks every time it runs.

It should also install all necessary modules from library and import them automatically.

Run this script from any endpoint and set up task schedule to run it every X minutes or hours.

[Type here]

Notification emails looks like this

The screenshot shows an email from 'onmicrosoft.com' to 'Vlad Admin'. The subject is '⚠ Security Alert - Sign-In Detected for'. The email header includes 'Account Monitoring' and 'AM'. The body features a red header '⚠ SECURITY ALERT - ACCOUNT SIGN-IN DETECTED ⚠'. Below it is a table of detected sign-in details:

Account Name:	onmicrosoft.com
Email:	onmicrosoft.com
Time (UTC):	03/25/2025 11:32:14
Status:	Failed (Code: 50126)
IP Address:	
Location:	Radal, NO
Client App:	Browser
Device:	Unknown
Browser:	Chrome 134.0.0
Operating System:	Windows10
Sign-in ID:	27c38514-c512-4cc2-87f7-bd4f2f7f1000

A pink callout at the bottom says 'If this was not you, please contact your system administrator immediately!'. At the bottom are 'Reply' and 'Forward' buttons.

Feel free to customize it / tweak it as you wan 😊