**Auto-Creating Folders & Assigning Permissions to Logged-In User**

**Purpose**

This PowerShell script is designed to:

1. Automatically detect the **currently logged-in user** on a Windows device (even when the script runs in **System context**, like in Intune).
2. Ensure two specific folders (C:\Oracle\_32 and C:\Oracle\_64) exist — creating them if they don’t.
3. Grant **Full Control** (NTFS permissions) to the currently logged-in Azure AD user on both folders.

**Use Case**

This script is particularly useful when:

* You are deploying **Oracle Instant Client** or other tools that require write access to specific folders.
* You're using **Microsoft Intune** to deploy apps or scripts, and the script runs as **System**, but needs to **grant folder access to the actual logged-in user**.
* You want to automate folder creation and permission setting during device provisioning or app installation.

**How the Script Works (Step-by-Step)**

1. **Detect Logged-In User**

Uses Get-CimInstance -Class Win32\_ComputerSystem to identify the **interactive session user**.

This works even when the script is running under SYSTEM, which is default in Intune.

1. **Resolve User UPN (Email)**

Looks up the user's **Azure AD email address (UPN)** from the registry path:

HKLM:\SOFTWARE\Microsoft\IdentityStore\LogonCache

This ensures folder permissions are assigned to the **correct Azure AD identity**, not just a local display name.

**Folder Creation**

* Checks for existence of the folders: C:\Oracle\_32 and C:\Oracle\_64.
* Creates them if they don't exist.

**Set NTFS Permissions**

* Grants **FullControl** permissions to the logged-in user using:

System.Security.AccessControl.FileSystemAccessRule

**Benefits**

* **Offline-compatible**: No need for AzureAD PowerShell modules or Graph API.
* **Safe for Intune**: Works in **System context**, ideal for scripts pushed via Intune.
* **Robust permission handling**: Uses UPN/email for accurate ACL assignment.
* **Reusable**: Can be extended to handle more folders, users, or access levels.

**Tested With**

* Microsoft Intune (PowerShell script & Win32 App deployment)
* Azure AD joined Windows 10/11 devices
* SYSTEM and user context

**Permission Granted**

* **FullControl**:

Read, Write, Modify, Execute, Delete – everything

Applied to all files and subfolders (ContainerInherit,ObjectInherit)

**Example Log Output**



