

# Exchange Online

This Book Section holds Exchange Online related documentations

- [Outlook Desktop - Repeated Sign-In Prompt \(WAM BrokerPlugin Reset\)](#)
- [Google Workspace to Exchange Online Mail Migration](#)

# Outlook Desktop - Repeated Sign-In Prompt (WAM BrokerPlugin Reset)

Field	Details
Document Type	Known Issue - Workaround
Applies To	Microsoft Outlook Desktop (Classic), Windows 10/11, Microsoft 365 / Exchange Online
Audience	2nd Line Support / Exchange Online Admins
Author	AK. Udofeh
Last Updated	February 2026

## Overview

This article documents a workaround for a recurring issue where Outlook Desktop prompts users to sign in repeatedly — as frequently as every 15 minutes to every hour - without retaining the session after completing authentication. This issue can affect individual users or a large number of users simultaneously across an organisation.

## The Issue

Users open Outlook and are presented with a "**Sign In**" prompt unexpectedly. After signing in successfully, the prompt reappears again after a short period. The following self-service steps do **not** resolve the issue:

- Restarting the laptop or PC
- Signing out of Outlook and signing back in
- Restarting Microsoft Teams or other Office apps

The issue may also surface across other Microsoft 365 services (Teams, OneDrive, SharePoint) simultaneously, as they share the same underlying authentication component.

## Root Cause

Modern Microsoft 365 applications do not handle authentication directly. Instead, they delegate all sign-in and token management to a Windows OS component called **Web Account Manager (WAM)** and its associated background plugin: `Microsoft.AAD.BrokerPlugin`.

## Normal Authentication Flow

User opens Outlook  
↓  
Outlook requests a token from WAM  
↓  
WAM calls Microsoft.AAD.BrokerPlugin  
↓  
Plugin communicates with Microsoft Entra ID  
↓  
Entra ID returns an OAuth access token (valid ~1 hour)  
and a refresh token (valid up to 90 days)  
↓  
WAM silently refreshes the token in the background  
when it expires - user is never prompted again

## What Breaks the Flow

When the `Microsoft.AAD.BrokerPlugin` folder becomes corrupted or enters a broken state, the silent background refresh fails. When the 1-hour access token expires, Outlook falls back to prompting the user because WAM cannot silently obtain a new one.

## Known Triggers

This corruption can occur due to, but is not limited to:

**Windows OS cumulative/security updates** (confirmed trigger - February 2026 Patch Tuesday, build 10.0.29510.1001 and earlier - January 2026 KB5074109)

Incomplete or interrupted Microsoft 365 app updates

User profile corruption

Mid-session password or MFA changes in Entra ID

Entra ID token store inconsistencies after account modifications

This workaround applies to all of the above scenarios. If a user is experiencing the repeated Outlook sign-in prompt and no Conditional Access policy or service health incident is identified as the cause, resetting the BrokerPlugin is the recommended first-line fix.

## Before You Start

Rule out the following before applying this fix at scale:

Check	Where
Active Microsoft service incident	M365 Admin Center > Health > Service Health
Recently modified Conditional Access policy	Entra Admin Center > Protection > Conditional Access
User password recently expired or changed	Entra Admin Center > Users
Entra sign-in logs showing policy interrupts	Entra Admin Center > Monitoring > Sign-in Logs

If none of the above are present, proceed with the fix below.

## Fix: Manual Steps (Per Machine)

Run on the affected user's machine, logged in as that user. You do not need to be a local admin for Steps 1-4.

## Step 1: Close All Microsoft Applications

Fully close **Outlook**, **Teams**, **OneDrive** and any other Office applications. Check the system tray and ensure none are still running in the background.

## Step 2: Navigate to the BrokerPlugin Folder

Press `win + R` and enter:

```
%localappdata%\Packages
```

Locate the following folder:

```
Microsoft.AAD.BrokerPlugin_cw5n1h2txyewy
```

## Step 3: Rename the Folder

Rename the folder by appending `.old` to the end:

```
Microsoft.AAD.BrokerPlugin_cw5n1h2txyewy.old
```

Do not delete the folder. Renaming it preserves a backup and instructs Windows to recreate it fresh on next boot. If you encounter the error "The action can't be completed because the folder or a file in it is open in another program", see the Troubleshooting section below before continuing.

## Step 4: Clear Cached Credentials

Open **Control Panel** ? **Credential Manager** ? **Windows Credentials**

Remove all entries beginning with:

```
MicrosoftOffice16_*
```

```
msteams_*
```

```
Microsoft_OC_*
```

Any entry referencing `office.com` or `outlook.com`

## Step 5: Reboot the Machine

Perform a **full restart** (not sign out). Windows will automatically recreate the `Microsoft.AAD.BrokerPlugin` folder in a clean state on boot.

## Step 6: Sign Back Into Outlook

Open Outlook. When prompted, sign in once. The session should persist without re-prompting.

## Fix: Automated Script (Bulk Deployment)

Use this script for fleet-wide remediation via RMM tool, Intune Remediation Script, or GPO logon script.

**Important:** This script must run in the affected user's context, not as SYSTEM or a local admin account. The BrokerPlugin folder is per-user profile.

```
# Step 1: Kill processes holding the BrokerPlugin before rename
Stop-Process -Name "Microsoft.AAD.BrokerPlugin" -Force -ErrorAction SilentlyContinue
Stop-Process -Name "backgroundTaskHost" -Force -ErrorAction SilentlyContinue
Stop-Process -Name "RuntimeBroker" -Force -ErrorAction SilentlyContinue

# Brief pause to allow process termination to complete
Start-Sleep -Seconds 2

# Step 2: Rename the BrokerPlugin folder
$brokerPath = "$env:LOCALAPPDATA\Packages\Microsoft.AAD.BrokerPlugin_cw5n1h2txyewy"

if (Test-Path $brokerPath) {
    try {
        Rename-Item -Path $brokerPath -NewName "$brokerPath.old" -Force -ErrorAction Stop
        Write-Output "SUCCESS: BrokerPlugin reset complete. Reboot required."
    }
    catch {
        Write-Output "ERROR: Failed to rename BrokerPlugin folder. $($_.Exception.Message)"
    }
} else {
    Write-Output "INFO: BrokerPlugin folder not found or already reset. No action taken."
}
```

**Post-deployment:** Push a reboot policy immediately after the script runs, or instruct users to restart when prompted.

## Troubleshooting — "The Action Can't Be Completed" Rename Error

If Windows displays:

*“The action can't be completed because the folder or a file in it is open in another program”*

This means a background Windows process is still holding a lock on the folder even though all visible apps are closed. The `Microsoft.AAD.BrokerPlugin` process runs silently in the background and can respawn quickly. Use the following method to identify and terminate it before retrying the rename:

Using Windows Resource Monitor (No Additional Tools Required)

Press **Win + R** and run:

```
perfmon.exe /res
```

In Resource Monitor, select the **CPU** tab

Scroll down and expand the **Associated Handles** section

In the search box, type:

```
AAD.BrokerPlugin
```

Press **Enter** — Resource Monitor will display all processes currently holding a handle on the folder

Right-click each result ? **End Process**

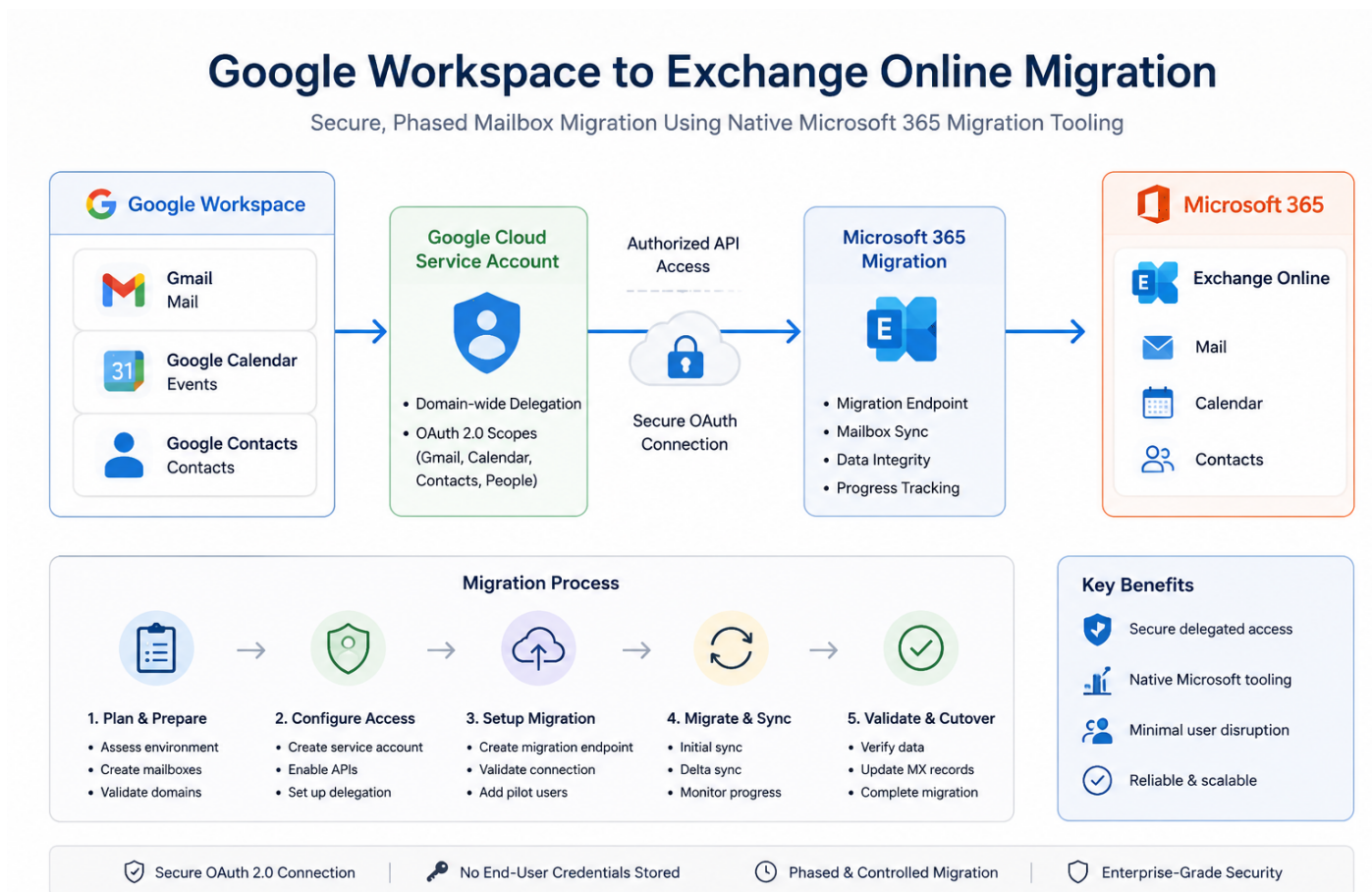
Immediately return to the folder and complete the rename (Step 3 above) before the process respawns

## Expected Outcome

<b>Resolution time</b>	Under 5 minutes + reboot
<b>User impact</b>	One-time sign-in prompt after reboot, then session persists normally
<b>Recurrence</b>	Should not recur unless a subsequent update re-introduces the regression

# Google Workspace to Exchange Online Mail Migration

Field	Details
Document Type	Google Workspace to Exchange Online Mail Migration
Applies To	Exchange Online, Google Workspace, Google API, Google Cloud
Audience	Systems Administrator / IT Engineer
Author	AK. Udofeh
Last Updated	May 2026



## Overview

This configuration enables mailbox migration from Google Workspace (Gmail) to Microsoft Exchange Online using the native migration functionality built into the Exchange Admin Center

(EAC).

The migration process uses a Google Cloud service account with delegated access to securely read Gmail, Calendar, and Contacts data from Google Workspace and import it into Microsoft 365 mailboxes.

This approach is important because it:

- Enables centralized migration management from Microsoft 365
- Supports staged or pilot migrations
- Minimises manual mailbox export/import operations
- Preserves mail, calendars, and contacts during migration

The configuration mitigates risks associated with:

- Manual PST exports
- Incomplete mailbox migrations
- Credential sharing
- Unsecured mailbox access methods

## Prerequisites

### Required Licenses

#### Microsoft 365

- Exchange Online Plan 1 or higher
- Microsoft 365 Business Premium / E3 / E5 recommended

#### Google Workspace

- Google Workspace Business or Enterprise subscription
- Super Admin access required

### Required Roles & Permissions

#### Microsoft 365

The administrator performing the migration requires:

- Exchange Administrator  
or
- Global Administrator

#### Google Workspace

The administrator requires:

- Super Admin role

## Dependencies

The following services must be accessible:

- Exchange Online
- Google Workspace Admin Console
- Google Cloud Console

## Preparation Tasks

Before beginning:

- Create Microsoft 365 mailboxes for all users being migrated
- Verify domains in Microsoft 365
- Ensure users have Exchange Online licenses assigned
- Confirm mailbox sizes and available storage
- Plan migration window and user communication

## Step 1: Configure Google Cloud Service Account

### Create Google Cloud Project

Navigate to:

<https://console.cloud.google.com/>

Create a new project.

Example:

*M365Migration*

### Create Service Account

Navigate to:

*IAM & Admin → Service Accounts*

Select:

*Create Service Account*

Example service account name:

*exchange-migration*

Select:

- Create
- Done

## **Enable Domain-Wide Delegation**

Open the newly created service account.

Navigate to:

*Details > Show Domain-wide Delegation*

Enable:

*Enable Google Workspace Domain-wide Delegation*

Enter a product name:

*Exchange Migration*

Save the configuration.

## **Record the Client ID**

Within the service account:

- Copy the Unique ID / Client ID
- Save it securely

This ID will later be used for delegated access configuration.

## **Create JSON Key**

Navigate to:

*Keys > Add Key > Create New Key*

Select

- JSON

Download and securely store the JSON key file.

Treat this file as sensitive credential material.

## Step 2: Configure Google Workspace Delegated Access

Navigate to:

*Google Admin Console > Security > Access and Data Control > API Controls*

Select:

*Manage Domain Wide Delegation*

Select:

*Add New*

### Configure Delegated Access

#### Client ID

Paste the service account Client ID copied earlier.

#### OAuth Scopes

Enter the following scopes exactly as shown:

*<https://mail.google.com/>,<https://www.googleapis.com/auth/calendar>,<https://www.google.com/m8/feeds/>,<https://www.googleapis.com/auth/gmail.settings.sharing>,<https://www.googleapis.com/auth/contacts>*

#### Important:

- Do not add spaces
- Use comma-separated format only
- Incorrect scopes will cause migration failures later

Select:

*Authorize*

### Step 3 — Enable Required Google APIs

In the Project page, navigate to:

<https://console.cloud.google.com/apis/library>

Ensure the correct migration project is selected.

Click Enable API Services and enable the following APIs:

API	Required
Gmail API	Yes
Google Calendar API	Yes
Contacts API	Yes
People API	Yes

## Step 4: Configure Migration Endpoint in Exchange Online

Navigate to:

*Exchange Admin Center > Migration*

Select:

*Add Migration Batch*

### Migration Path

Choose:

*Google Workspace (Gmail)*

### Migration Endpoint Configuration

#### Email Address

Enter a Google Workspace Super Admin account.

Example:

*admin@company.com*

Do not use the service account email address.

### JSON Key File

Upload the downloaded JSON key file created earlier.

### Verification

If endpoint validation repeatedly fails:

- Enable Skip Verification
- Continue with pilot migration testing

Google propagation delays may cause temporary validation failures.

## **Step 5: Access Control / Enforcement**

### **Recommended Migration Scope**

For production safety:

- Begin with pilot users only
- Avoid immediate tenant-wide migration

Recommended pilot group:

- IT administrators
- Test users
- Low-risk business users

### **Recommended Mail Flow Strategy**

During migration:

- Keep Google Workspace as primary mail delivery platform
- Do not switch MX records immediately

Switch MX records only after:

- Mailbox validation
- User acceptance testing
- Successful pilot migration completion

## **Step 6: Testing / Report Mode**

### **Recommended Pilot Migration Process**

Migrate:

- One mailbox initially
- Validate data integrity
- Confirm permissions and access

### **Validate Migrated Data**

Confirm:

- Emails migrated successfully
- Folder structure preserved
- Calendar items imported
- Contacts available
- Outlook access functional

## User Validation

Perform:

- Outlook sign-in testing
- OWA testing
- Mobile device testing

## Step 7: Monitoring & Validation

### Exchange Online Monitoring

Navigate to:

Exchange Admin Center > Migration

Monitor:

- Batch status
- Sync progress
- Failed items
- Skipped items

### Google Workspace Validation

Validate:

- API access remains active
- Service account remains enabled
- Delegation settings remain configured

### Common Issues to Monitor

Issue	Likely Cause
Endpoint validation failure	Propagation delay
Authentication failure	Incorrect OAuth scopes
Mailbox sync failure	API not enabled
Permission denied	Delegation not configured

## Step 8: Enforcement / Go-Live

Once migration validation is complete:

### Finalize Migration

Complete:

- Final synchronization
- User sign-off
- Mail flow cutover

### Update MX Records

Point MX records to Microsoft 365.

Example Microsoft MX target:

*<tenant>.mail.protection.outlook.com*

### Post-Cutover Tasks

Perform:

- Outlook profile validation
- Mobile device reconfiguration
- DNS validation
- Mail flow testing

## Important Considerations

### Propagation Delays

Google delegation and API changes may take:

- 15 minutes to 24 hours

Temporary failures during this period are expected.

### Service Account Security

The JSON key file provides privileged access.

Recommendations:

- Store securely
- Restrict administrator access
- Delete unused keys after migration

## Verification Failures

Microsoft endpoint verification may intermittently fail even when configuration is correct.

Where necessary:

- Use Skip Verification
- Validate with pilot migrations

## Large Mailboxes

Large Gmail mailboxes may:

- Take several hours
- Experience throttling
- Require staged synchronization

## Best Practices

### Security Recommendations

- Use dedicated migration admin accounts
- Restrict service account access
- Remove unused delegation after migration completion
- Rotate or delete JSON keys post-migration

### Operational Recommendations

- Start with pilot users
- Avoid weekend cutovers without validation
- Maintain rollback capability during migration
- Document all DNS changes

### Migration Recommendations

- Migrate mailboxes in batches
- Validate each batch before proceeding
- Communicate migration timelines clearly to users

## Summary

This implementation configured secure mailbox migration from Google Workspace to Exchange Online using Microsoft's built-in Google Workspace migration functionality.

The process included:

- Google Cloud service account creation
- API enablement
- Domain-wide delegation
- Exchange Online migration endpoint configuration
- Pilot migration validation
- Controlled production rollout

Following this approach provides a secure, enterprise-ready migration process while minimising disruption, authentication issues, and mailbox migration failures.