



# Intune Deep Dive: How Policy And App Delivery Really works...

# Sponsors





Rudy Ooms

Microsoft MVP 5 years  
Patch My PC

Multiple Nicknames

- TroubleMaker
- The guy that reverse Code for fun
- Mister DLL
- Mister MDM

Hobbys? → The Above





**\*Joost Gelijsteen**

Microsoft MVP 4 years  
Secure at Work  
Hobbys Sneaker, Sport  
\*It's like toast... but with a J



# The Goal of this Webinar



Giving People  
a clear understanding  
How policies  
coming down...



How it works...  
How it doesn't Work!



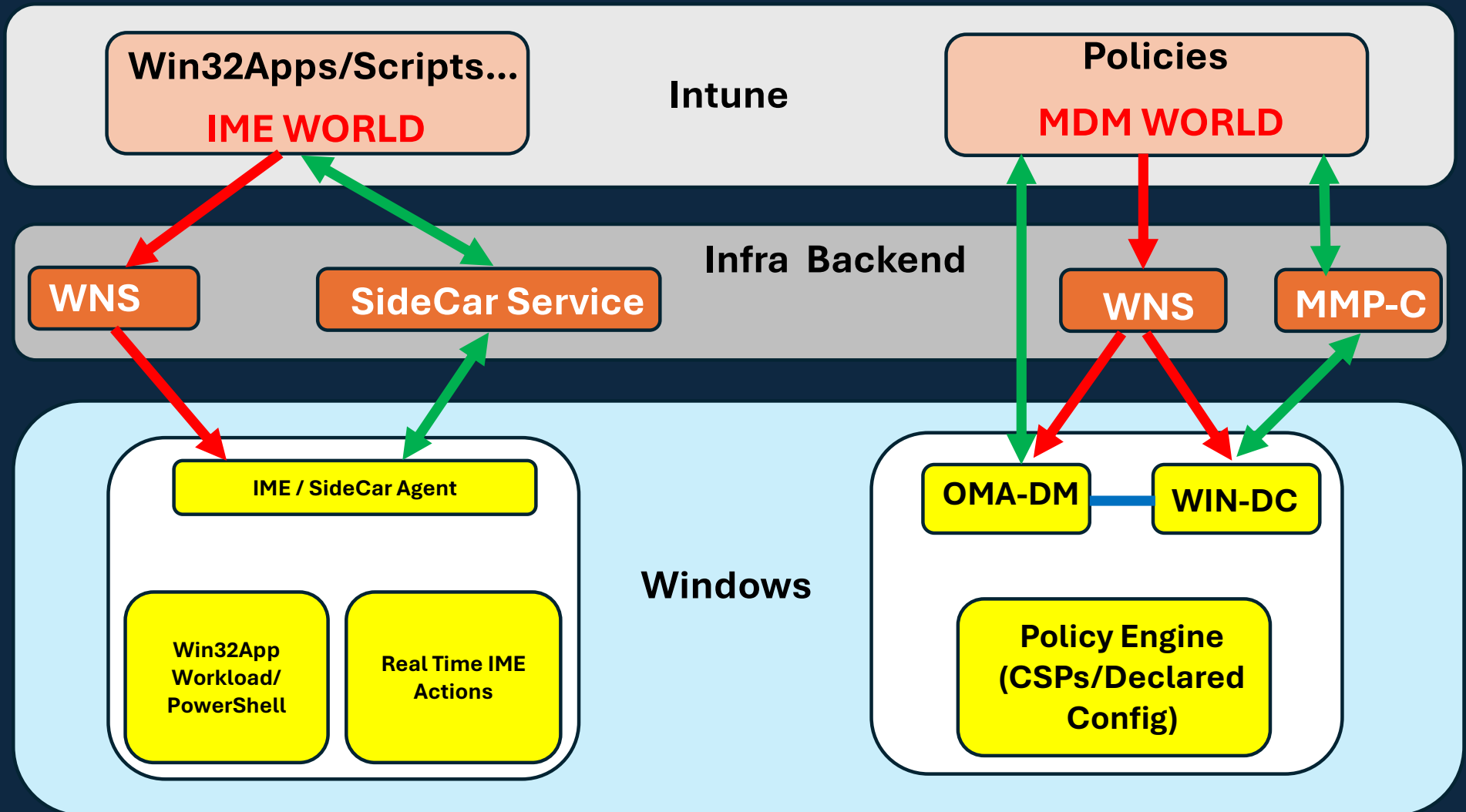
How Microsoft is  
working on  
Improvements

# Agenda

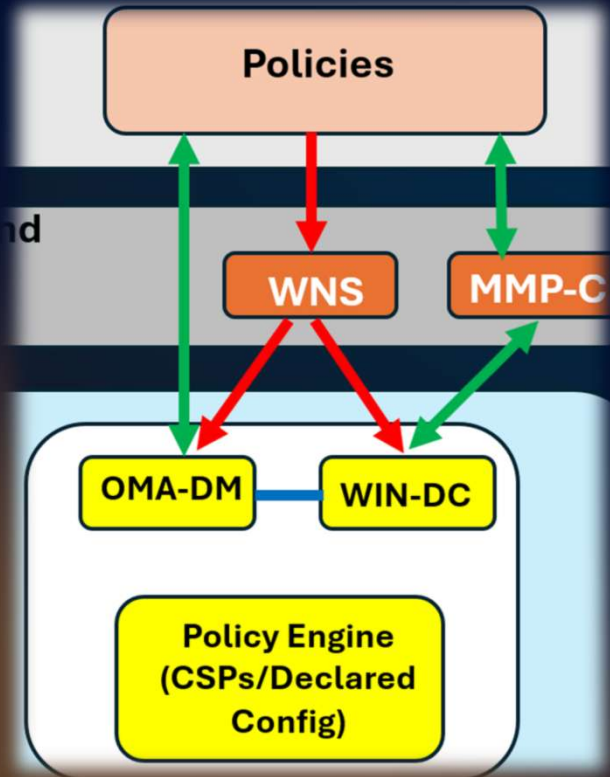


- The Basics
- The Full Policy Flow
- Windows Notification Service
- The Fastlane
  - Maybe More... ?(Depending on Time)

# The Basics



# How Intune policies are delivered



Let's first  
examine  
how and  
when  
**policies** are  
sent over



# Deploying Policies



**Configuration settings**

Device Lock

- Device Password Enabled  Enabled
- Allow Simple Device Password  Allowed.
- Alphanumeric Device Password Required  Password, Numeric PIN, or Alphanumeric PIN required.
- Device Password Expiration  0
- Device Password History  0
- Max Device Password Failed Attempts  0
- Max Inactivity Time Device Lock  0
- Min Device Password Length  4

Scope tags

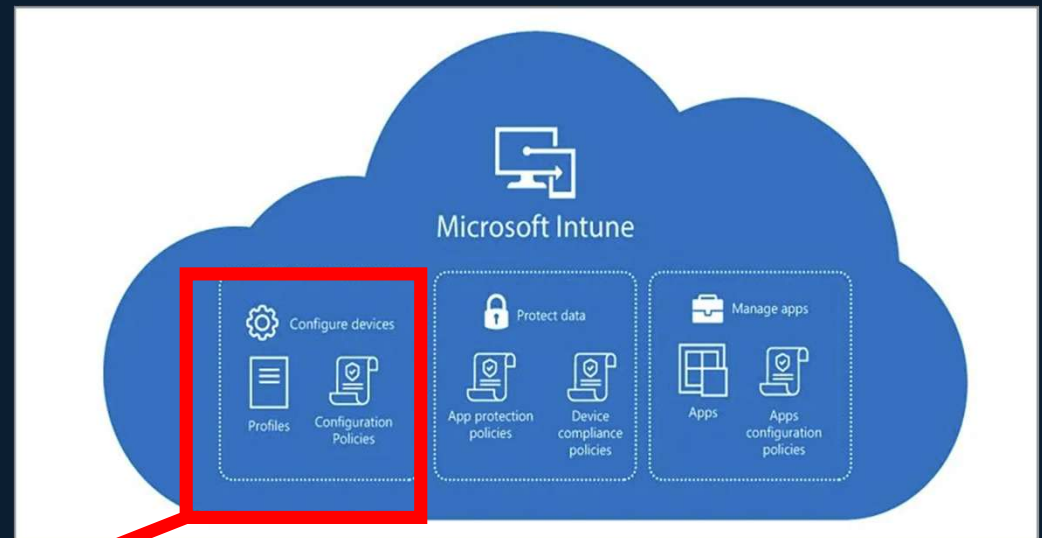
Default

Assignments

Included groups

Group	Status	Group Members	Filter
All devices	Active		None

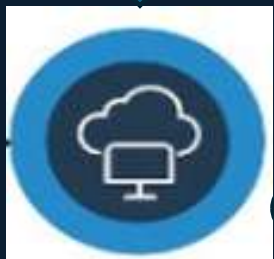
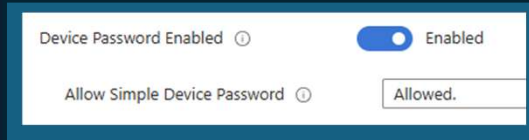
Excluded groups



Let's start creating a Policy

# How Intune policies are delivered

1 Configure Policy in Intune



2 Intune sends policies in SyncML (XML format).



3 MDM Client (OMA-DM) parses the SyncML payload.



4 The CSP engine applies it locally. (get / set / get)

- ./Device/Vendor/MSFT/Policy
  - Config
    - {AreaName}
    - {PolicyName}
  - ConfigOperations
    - ADMXInstall
      - {AppName}
      - {SettingsType}
      - {AdmxFileId}
    - Properties
      - {SettingsType}
      - {AdmxFileId}
      - Version

Let's walk you through every step!

# How Intune policies are delivered



How  
**policies** are  
sent over to  
the device  
...in detail

# What happens when creating a new policy



Configuration settings

Device Lock

Device Password Enabled  Enabled

Allow Simple Device Password  Allowed.

Alphanumeric Device Password Required  Password, Numeric PIN, or Alphanumeric PIN required.

Device Password Expiration  0

Device Password History  0

Max Device Password Failed Attempts  0

Max Inactivity Time Device Lock  0

Min Device Password Length  4

Scope tags

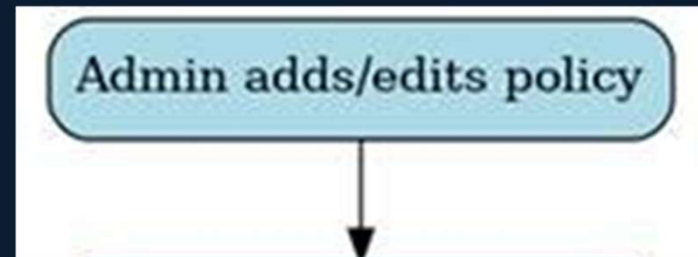
Default

Assignments

Included groups

Group	Status	Group Members <input type="text"/>	Filter
All devices	Active		None

Excluded groups



So... again... lets create the policy first...

## Do we need to wait?

PushLaunch	Ready	Custom Trigger		9/29/2025 1:16:2
PushRenewal	Ready	Multiple triggers ...	10/13/2025 10:39:19 PM	9/29/2025 7:39:5
Schedule #1 created by enrollment client	Ready	At 10:41 PM on 9...		9/29/2025 7:53:0
Schedule #2 created by enrollment client	Disabled	At 10:56 PM on 9...		9/29/2025 8:41:0
Schedule #3 created by enrollment client	Ready	At 12:56 AM on 9...	9/29/2025 5:56:00 PM	9/29/2025 9:56:0
Schedule created by enrollment client for r...	Ready	At 6:33 AM on 4/...	4/10/2026 3:33:00 PM	11/30/1999 12:0
Schedule to run OMADMClient by client	Ready			9/29/2025 1:16:4

General Triggers Actions Conditions Settings History (disabled)

When you create a task, you can specify the conditions that will trigger the task. To change these triggers, open the task property pages using the Properties command.

Trigger	Details	Status
One time	At 12:56 AM on 9/29/2025 - After triggered, repeat every 08:00:00...	Enabled

Do we need to wait 8-hours before the policy arrives at the device?

**NOPE... let us explain!**

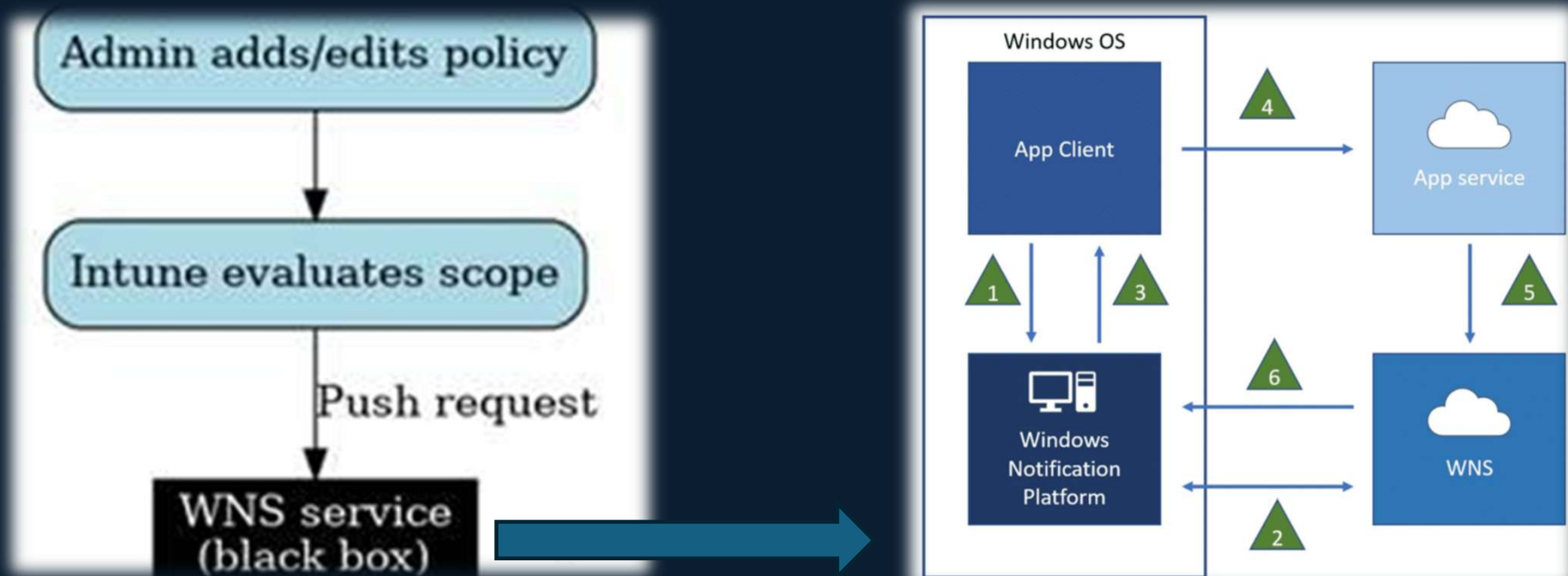
# WNS



**WNS** Is  
(still)  
pretty  
important  
(for now)

- When we are changing targeting (adding or removing a device/user group)
- When we are Editing a payload (changing/adding a new Intune policy)
- When Entra group membership changes

**Intune will ask the WNS to deliver a message to the device, telling it to check in.**



## WNS the Blackbox

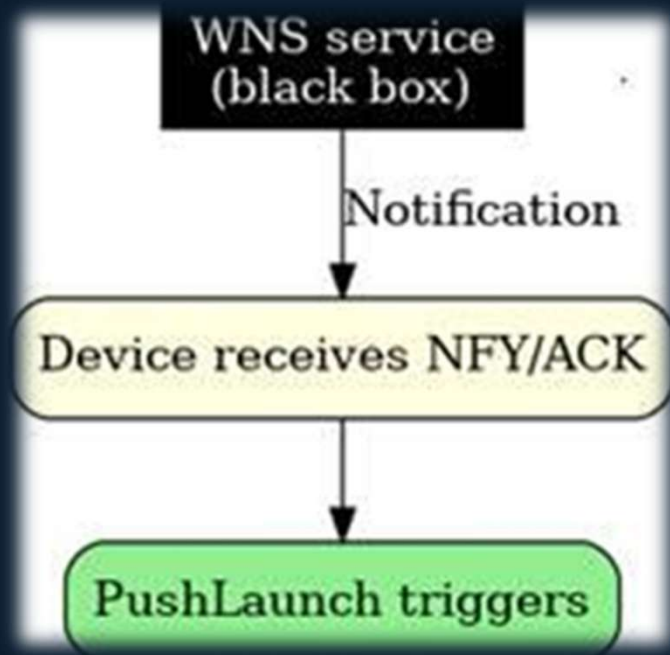
WNS “**should**” send a message to device asking it to check in..



But with WNS being a black box. That message might reach the device instantly, or it might disappear into the void without you ever knowing.



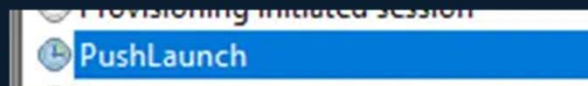
# The Push Launch Task



The Push Message is just a wake-up call!

```
WNP Transport Layer received command: PNG, Trid: 31, Namespace: CON, CV: RVK4dyN4DEGtvrV1.22.0 containing 126 bytes of payload: 0x4D532D43563A2052564B3464794E3444454774767256312E32322E300D0A0D0A3C70696E672D726573706F6E73653E3C776169743E34393C2F776169743E3C636F6E6E656374696F6E2D7374617475733E436F6E6E65637465643C2F636F6E6E656374696F6E2D7374617475733E3C2F70696E672D726573706F6E73653E. IsLongRunning: Data Connection.
```

Once and if the push message arrives, It wakes up the PushLaunch scheduled task.



How does the PUSH wake up the PushLaunch Task?



# The Push Launch Task

The PushLaunch task "listens" to a specific (75e0....) Windows Notification Facility (WNF) State Change

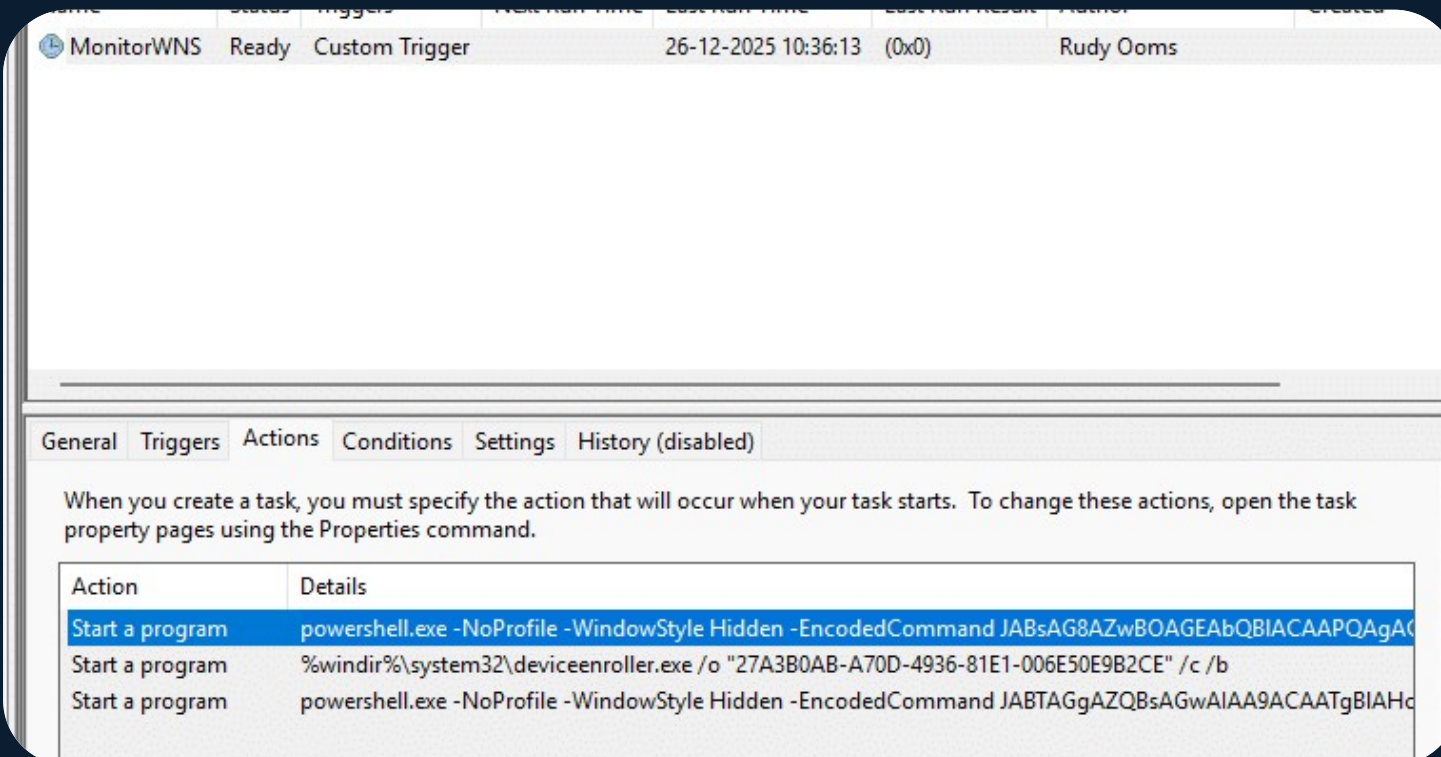
Something Happened..  
And now we need to take action

```
</Settings>  
<UseUnifiedSchedulingEngine>true</UseUnifiedSchedulingEngine>  
</Settings>  
<Triggers>  
  <WnfStateChangeTrigger>  
    <StateName>75E0BEA328009213</StateName>  
  </WnfStateChangeTrigger>  
</Triggers>  
<Actions Context="LocalSystem">  
  <Exec>  
    <Command>%windir%\system32\deviceenroller.exe</Command>  
    <Arguments>/o "92C1BE86-70F0-4839-A677-5DA4D84FEE86" /c /z</Arguments>  
  </Exec>  
</Action>
```

232 75A0BCA328009213 WNF\_ENTR\_PUSH\_RECEIVED An enterprise WNS based push was received

\*WNF is the notification system within the Windows OS

## The Push Launch Task



The screenshot shows a Windows Task Scheduler task named "MonitorWNS". The task is in a "Ready" state and has a "Custom Trigger". The trigger details show it is scheduled for "26-12-2025 10:36:13" with a duration of "(0x0)". The task was created by "Rudy Ooms".

The task properties are shown in the "Actions" tab. The instructions state: "When you create a task, you must specify the action that will occur when your task starts. To change these actions, open the task property pages using the Properties command."

Action	Details
Start a program	powershell.exe -NoProfile -WindowStyle Hidden -EncodedCommand JABsAG8AZwBOAGEAbQBIACAAPQAgAC
Start a program	%windir%\system32\deviceenroller.exe /o "27A3B0AB-A70D-4936-81E1-006E50E9B2CE" /c /b
Start a program	powershell.exe -NoProfile -WindowStyle Hidden -EncodedCommand JABTAGgAZQBAsAGwAIAA9ACAATgBIAHc

The funny thing is that you can create your own scheduled task with the same WNF State Change to trigger "something"  
Don't do that... Only Rudy Does that...


# OMA-DM Client Sync

PushLaunch triggers

OMADMClient syncs

```
loc_140019F95:  
test byte ptr cs:Microsoft_WindowsPhone_OmaDm_Client_ProviderEnableBits, 8  
jz short loc_140019FD7
```

3 MDM Client (OMA-DM) parses the SyncML payload.

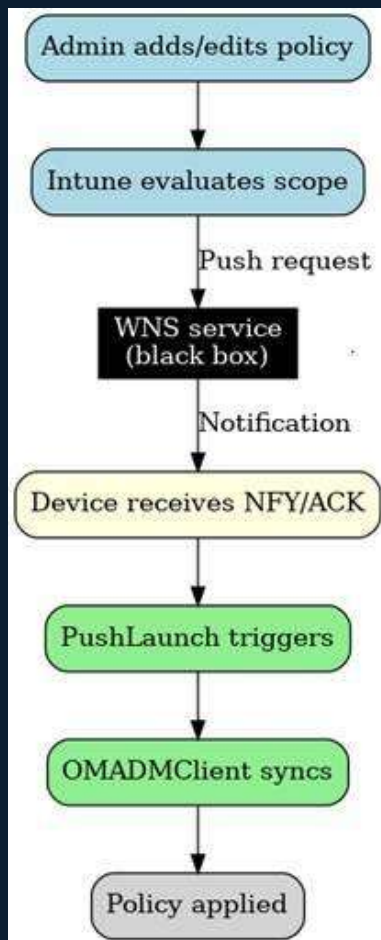


Schedule created by enrollment client for r...	Ready	At 6:33 /
Schedule to run OMADMClient by client	Ready	

The PushLaunch task will \*kick off The OMA-DM (phoneprovider) client and will reach out to Intune to fetch the policies

\*with a delay of 5 minutes Queued Schedule  
(if you don't have the feb 2026 update installed)

## How A device receives a policy in nutshell



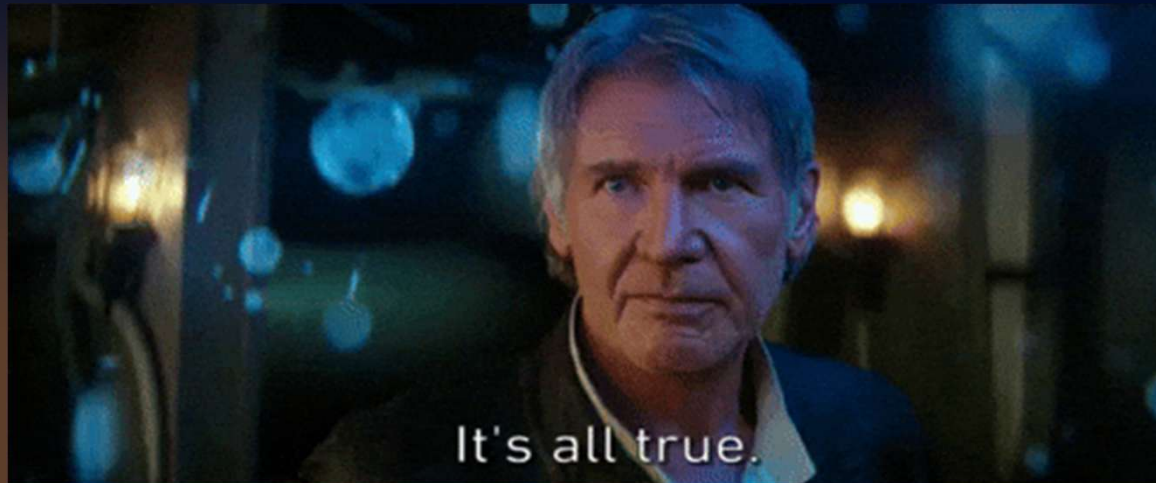
### Summary:

Editing/Adding a policy will ask WNS (the blackbox) to wake up the device.

It will trigger the **pushlaunch**, which will kick off the Sync with Intune to receive the policies.



We are not lying?



It's all true.

Demo to  
Prove it

Home > Devices

# Devices | Configuration

Search

- Overview
- All devices
- Device query
- Monitor
- By platform
  - Windows
  - iOS/iPadOS
  - macOS
  - Android
  - Linux
- Device onboarding
  - Windows 365
  - Enrollment
- Manage devices
  - Configuration**
  - Compliance
  - Conditional access
  - Scripts and remediations
  - Group Policy analytics
  - eSIM cellular profiles (preview)
  - Policy sets
  - Device categories
  - Partner portals

Policies Import ADMX Monitor

+ Create Refresh Export Columns


105 policies

Search Add filters

Policy name	Policy type	Last modified
Config Refresh	Settings catalog	1/29/2026, 8:22:56 AM
Allow Camera	Settings catalog	1/29/2026, 12:58:17 AM
Require Private Store Only Only Private store is enabled	Settings catalog	1/21/2026, 7:40:41 AM
Autopilot reset allow	Device restrictions	1/19/2026, 4:43:20 AM
Block local app reset	Settings catalog	1/19/2026, 4:40:02 AM
Secure Boot	Settings catalog	1/9/2026, 3:56:52 AM
Defender App Guard	App and Browser Isolation	1/4/2026, 5:07:00 AM
AV Tamper Protection	Windows Security Experience	1/4/2026, 5:06:31 AM
BL-WIN-USR-P-AV-MDE Security Baseline v24H1-Microsoft D	Microsoft Defender Antivirus	12/30/2025, 4:19:41 AM
properties	Properties catalog	12/24/2025, 2:57:54 AM
Maintenance	Device restrictions (Windows 10 Team)	12/14/2025, 10:48:33 PM
Windows Autopatch Microsoft 365 Apps Update Policy - Aut	Settings catalog	12/14/2025, 10:34:38 PM
Windows Autopatch Microsoft 365 Apps Update Policy - Aut	Settings catalog	12/14/2025, 10:34:38 PM
Windows Autopatch Edge Update Policy - Autopatch - Test	Settings catalog	12/14/2025, 10:34:38 PM
Windows Autopatch Edge Update Policy - Autopatch - Last	Settings catalog	12/14/2025, 10:34:37 PM
Windows_DO	Settings catalog	12/9/2025, 3:06:10 AM
BL-WIN-USR-P-DEVCONF-MDM Security Baseline v23H2-SC	Settings catalog	12/9/2025, 3:04:28 AM
EndpointProtection_ExploitProtection	Exploit Protection	12/9/2025, 2:51:51 AM

https://intune.microsoft.com/#view/Microsoft\_Intune\_DeviceSettings/DevicesMenu/~/\_allDevices

# The 8 hour sync myth Busted



PushLaunch	Ready	Custom Trigger		9/29/2025 1:16:2
PushRenewal	Ready	Multiple triggers ...	10/13/2025 10:39:19 PM	9/29/2025 7:39:5
Schedule #1 created by enrollment client	Ready	At 10:41 PM on 9...		9/29/2025 7:53:0
Schedule #2 created by enrollment client	Disabled	At 10:56 PM on 9...		9/29/2025 8:41:0
Schedule #3 created by enrollment client	Ready	At 12:56 AM on 9...	9/29/2025 5:56:00 PM	9/29/2025 9:56:0
Schedule created by enrollment client for r...	Ready	At 6:33 AM on 4/...	4/10/2026 3:33:00 PM	11/30/1999 12:0
Schedule to run OMADMClient by client	Ready			9/29/2025 1:16:4

General Triggers Actions Conditions Settings History (disabled)

When you create a task, you can specify the conditions that will trigger the task. To change these triggers, open the task property pages using the Properties command.

Trigger	Details	Status
One time	At 12:56 AM on 9/29/2025 - After triggered, repeat every 08:00:00...	Enabled

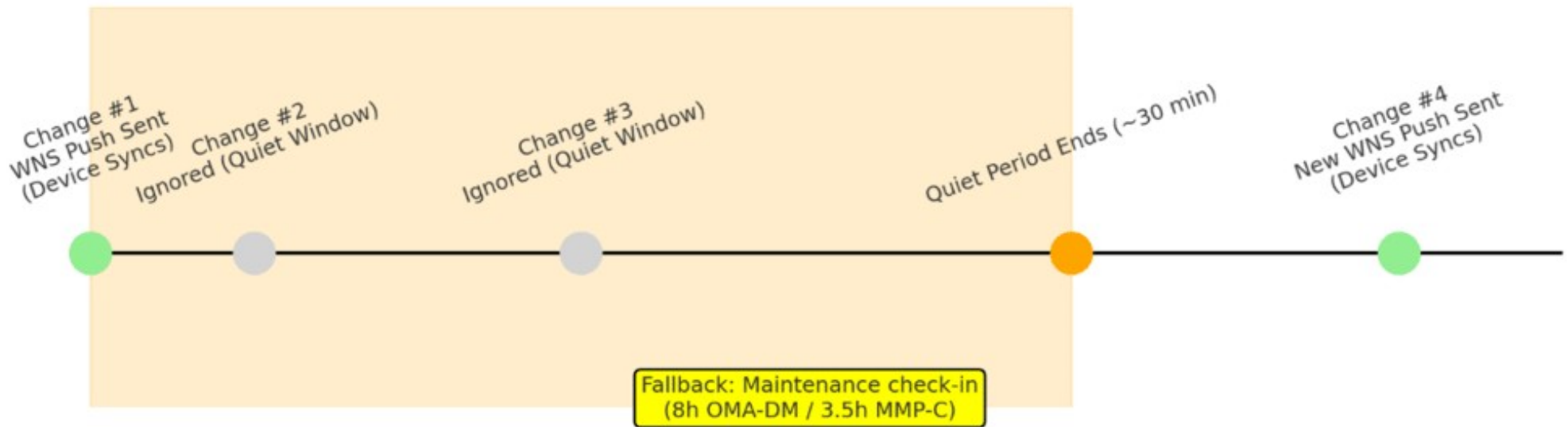
**So we don't need to wait or manually sync our devices  
(if the push.. Arrives..if.....)**

## Multiple Changes



What happens if we make **multiple changes?**

## Be aware of throttling!! (for now)



- Change #1 → WNS push almost immediate
- Change #2 and #3 (<30 min later) → bundled, resolved when device responds to the same notification
- Change #4 (>30 min later) → new WNS push delivered

**It's not the 8-hour sync... it's the 30-minute throttle window that's the real delay.**

That's where the new "Fast Lane" will improve things



## The Fast Lane (when its there!)

### The Fast Lane: How It Changes Policy Delivery



- When a drift (change) is detected, a timer starts.
- If no further changes happen within ~3 minutes, a notification is sent immediately.
- If more changes occur, the system queues them in small increments (up to ~10 minutes)
- So the 30 minutes Throttle will be gone!

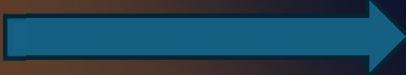
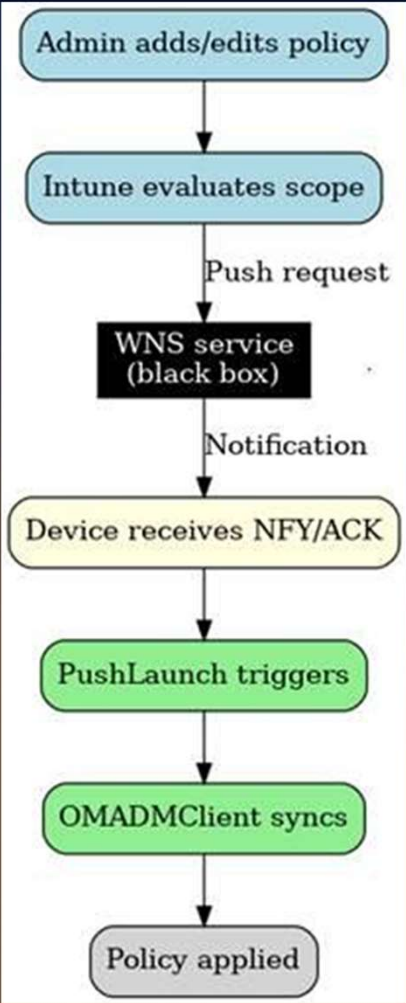
WNS did its job... what now?



How  
Policies Are  
being  
Applied



# Applying the Policies on the device

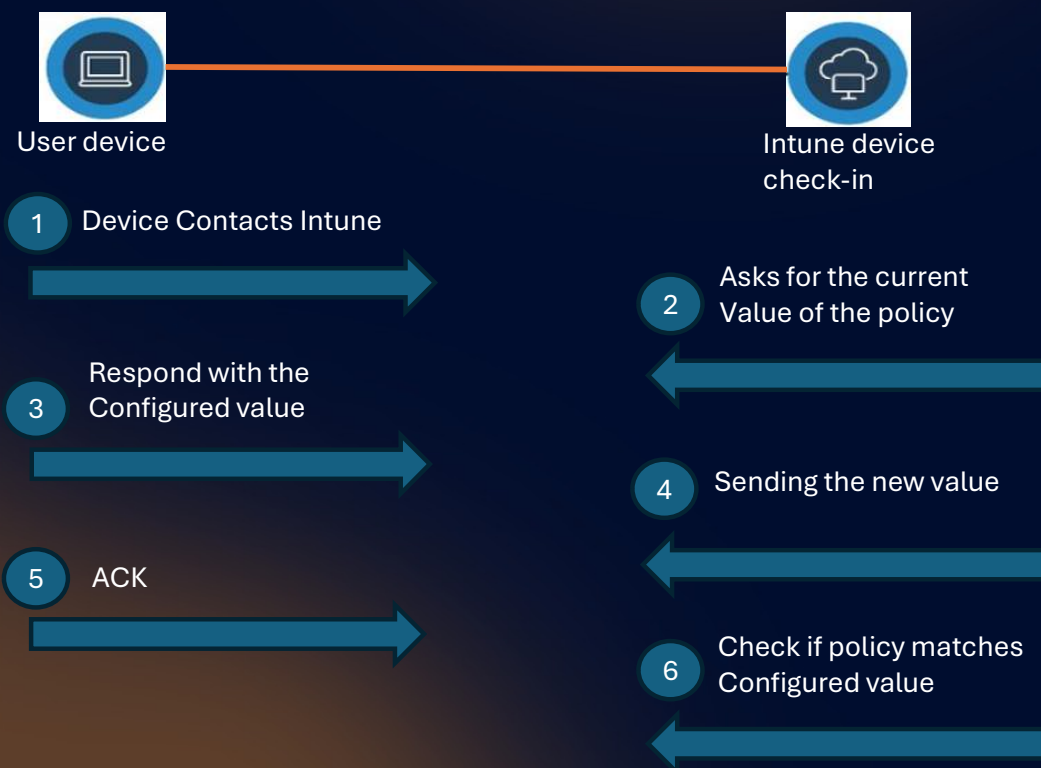


Now it's time for the Configuration Service Provider (CSP) to start applying the policies

4 The CSP engine applies it locally. (get / set / get)

- ./Device/Vendor/MSFT/Policy
  - Config
    - {AreaName}
    - {PolicyName}
  - ConfigOperations
    - ADMXInstall
      - {AppName}
      - {SettingsType}
      - {AdmxFileId}
    - Properties
      - {SettingsType}
      - {AdmxFileId}
      - Version

# Today's Enrollment & Policy Delivery



Policies delivered using OMA-DM Protocol (SyncML format)

Multiple round trips PER policy: **get** → **set** → **get**

# Today's Enrollment & Policy Delivery

Example: Setting the Config Refresh Cadence to 30 Minutes

(it was 45)

Get 45 / Set 30 / Get 30



## Edit profile - Config Refresh

Settings catalog

1 Configuration settings 2 Review + save

+ Add settings ⓘ

^ Config Refresh

Remove category

Refresh cadence \* ⓘ

30



Config refresh ⓘ

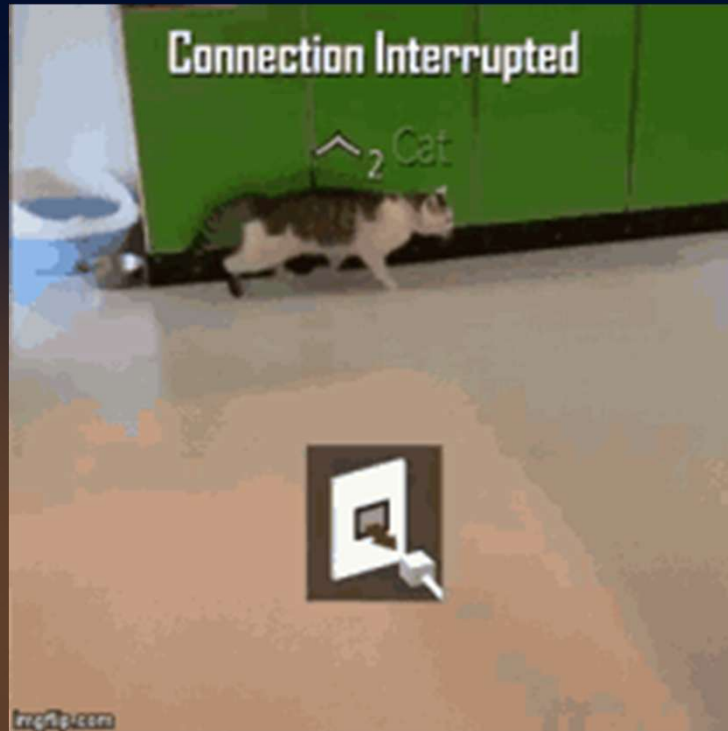


Enabled.





# Latency



Why  
Latency is  
NOT funny  
(well maybe a bit)

# The Latency Bottleneck



- Each policy requires multiple roundtrips (GET → SET → GET)  
→ **Slows down deployment and increases failures**
- Throttling kicks in after too many requests

(we have all been there... clicking the sync button a lot of times to speed it up)



→ **Devices may stop processing policies until throttle is gone**

- No offline enforcement  
→ **Devices can't correct or apply policies unless connected to the cloud.**
- Admins get limited visibility when something fails  
→ **Troubleshooting becomes reactive and time-consuming.**

Yes (msft) can!

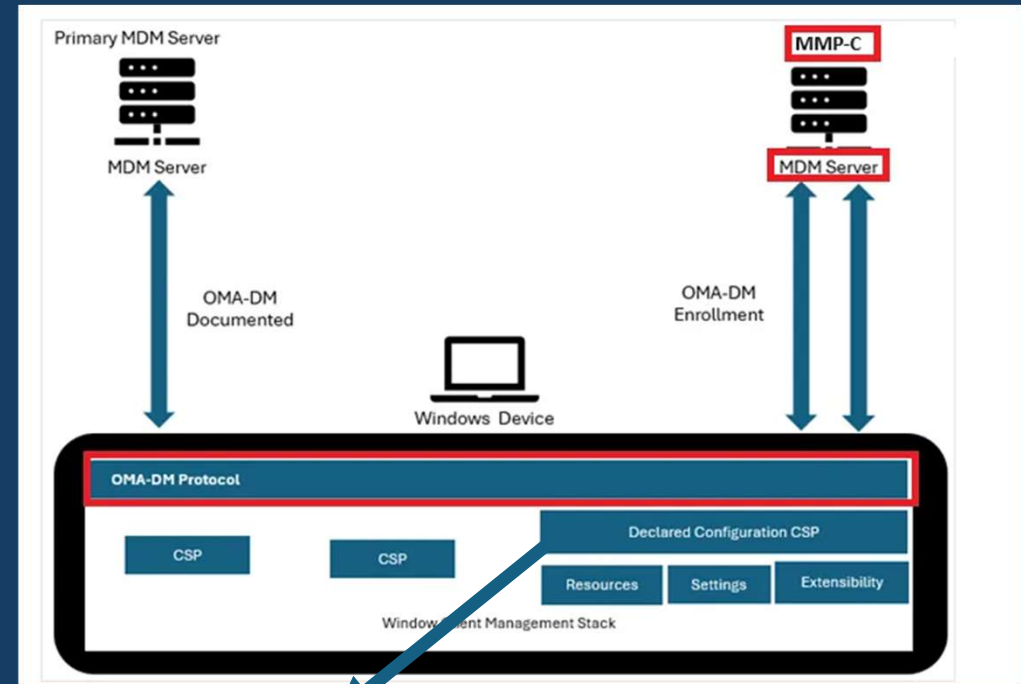
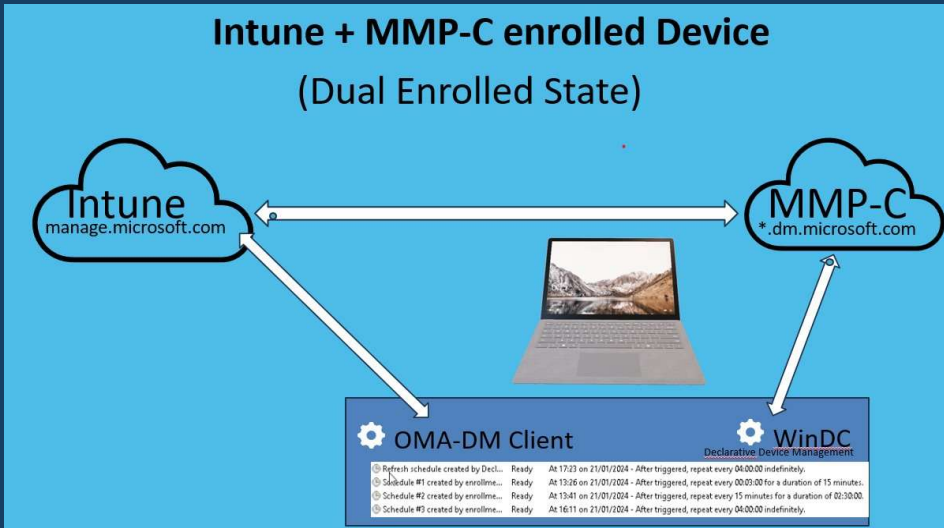


Fixing the  
Policy  
Latency

# This is called a Linked/Dual Enrolled

All Devices now have both  
OMA-DM and WinDC channel

Intune + MMP-C enrolled Device  
(Dual Enrolled State)

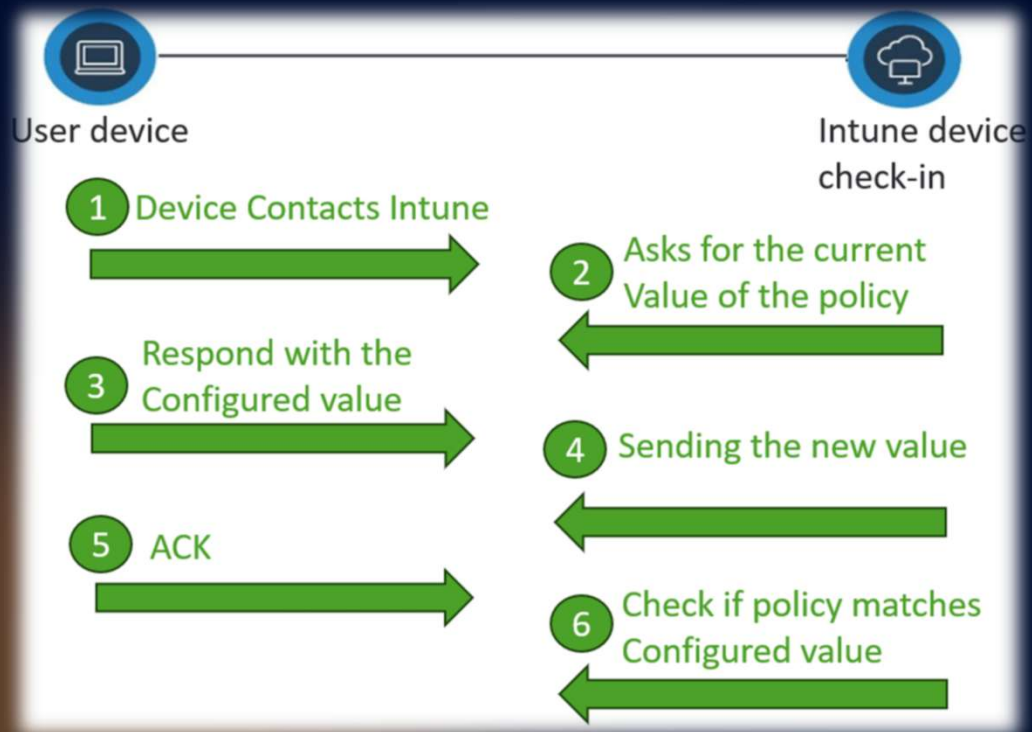


Which lets devices manage their own settings and send status updates to the management server on their own, without having to constantly check in



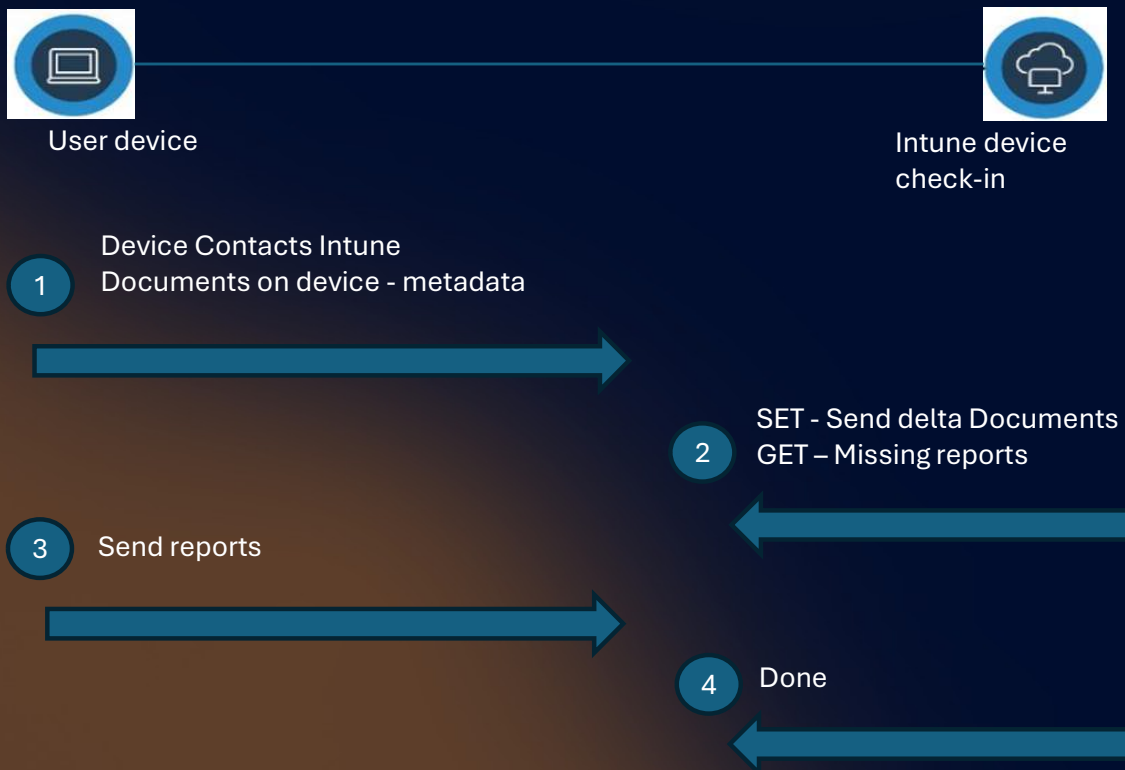
# Declared Configuration Policies

**Small Recap: The OLD Way**  
**From Get – Set – Get....for every policy (slow...)**



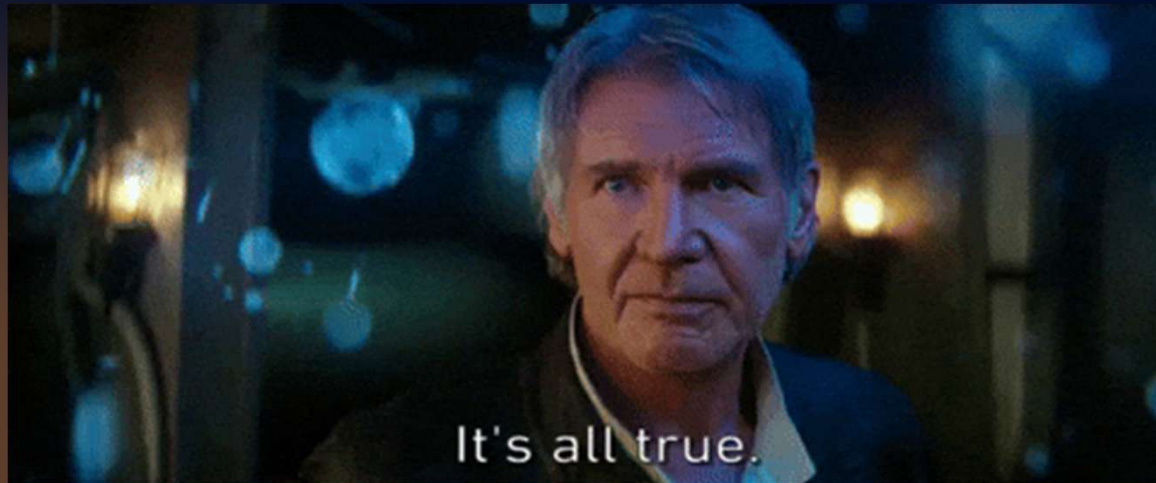
# Declared Policies

To SET / GET...for everything at once (The Fastlane!)



- batched declared configuration document that describes the entire desired state.
- Device will keep itself in a desired state Even offline!
- Will proactively checkin to intune if a change occurs

It's Fast.... We Are not Lying



Demo to  
Prove it

Computer\HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\DeclaredConfiguration\HostOS\Config\enrollments

Name	Type	Data
(Default)	REG_SZ	(value not set)

SyncML Viewer - oliverkieselbach.com - 1.4.0

File Options Actions Help

SyncML Representation Protocol Stream SyncML Sessions/Messages SyncML Requests Connectivity Profiles MDM Diagnostics About

1

Clear Stream Save As

HostOS Search HostOS

Details

7bfc795c237f8d331315ebbca4aa36cc3717dea6b7e4e794c0ec24a45f8a27f\_84BD4A16  
243c49278eaf46d66aa35aceae1f3a29f3aa14866d3af70817b42b017623719\_4EA48817  
af38b3b7e4f1b6f7fb287305e6f178f61ca0fea9339b8283f0243424bc87f\_2B5910C9-B3

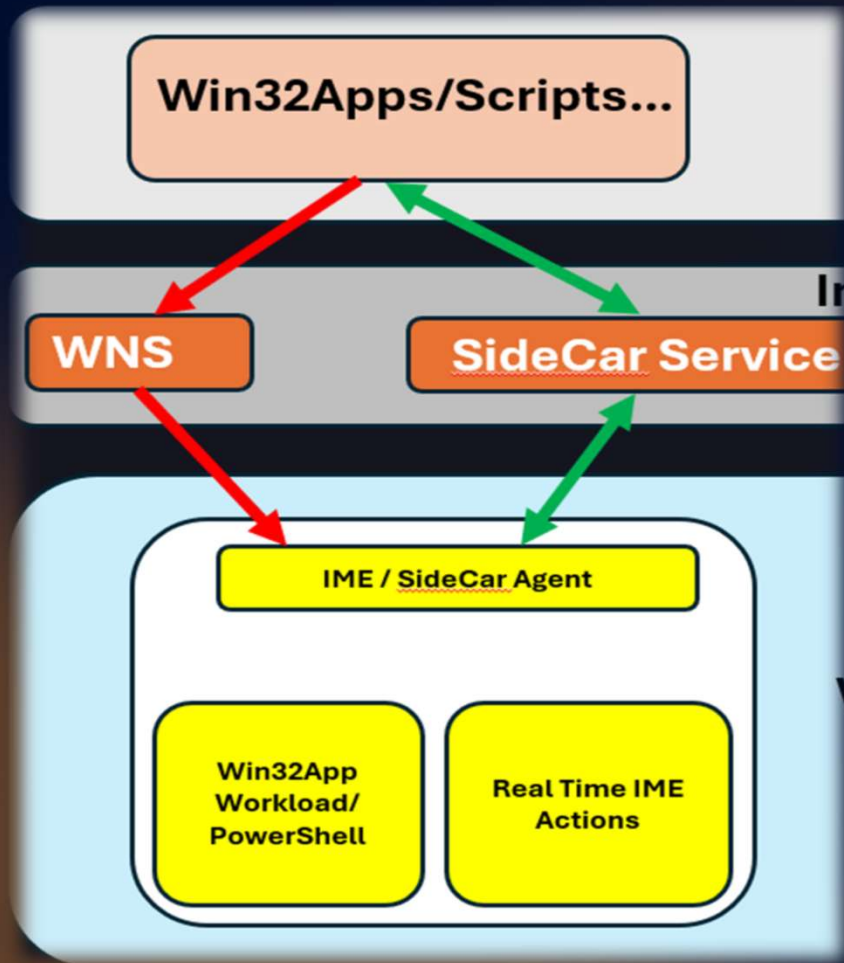
- ScenarioSchema
- DevDiv
- Device Association Framework
- DeviceInventory
- DeviceManageabilityCSP
- DeviceReg
- Dfrg
- DFS
- DiagnosticLogCSP
- DIAL
- DirectDraw
- DirectInput

DESKTOP-HRCEA00

Network

3 items

# What about the Apps/Scripts?



How the **IME / SideCar agent** is also evolving

# What about Apps?

```

public CheckinIntervalManager(ITimer timerWrapper)
{
    Guard.ArgumentNotNull((object) timerWrapper, nameof (timerWrapper));
    this.timerWrapper = timerWrapper;
    this.shouldUpdateTimer = false;
    this.intervalForTimer = 3600000.0;
    this.setTimerIntervalToDefaultFlag = false;
}

public void SetTimerInterval(int? nextCheckinIntervalMins)
{
    this.shouldUpdateTimer = true;
    this.intervalForTimer = !nextCheckinIntervalMins.HasValue ? 3600000.0 : (double) (nextCheckinIntervalMins.Value
    AppWorkloadLog.TraceInformation($"[Win32App][CheckinIntervalManager] SetTimerInterval. {${"shouldUpdateTimer"}");
}

public void SetTimerIntervalToDefault()
{
    this.setTimerIntervalToDefaultFlag = true;
    AppWorkloadLog.TraceInformation("[Win32App][CheckinIntervalManager] SetTimerIntervalToDefault.");
}

public void UpdateTimerIntervalForNextCheckin()
{
    AppWorkloadLog.TraceInformation($"[Win32App][CheckinIntervalManager] UpdateTimerIntervalForNextCh
    if (this.timerWrapper.GetInterval() == this.intervalForTimer)
        return;
    AppWorkloadLog.TraceInformation($"[Win32App][CheckinIntervalManager] {nameof (UpdateTimerInterval
    if (this.setTimerIntervalToDefaultFlag)
        this.timerWrapper.UpdateInterval(3600000.0);
    else if (this.shouldUpdateTimer)
        this.timerWrapper.UpdateInterval(this.intervalForTimer);
    AppWorkloadLog.TraceInformation($"[Win32App][CheckinIntervalManager] {nameof (UpdateTimerInterval
}

```

Today, required Apps and PowerShell Scripts  
Rely on **PULL** instead of **Push**

```

if (emsPolicyList1 != null && emsPolicyList1.Any<EmsPolicy>() && AADJoinHelper.IsDevic
{
    Log.TraceInformation("[PowerShell] Calculating effectivePolicies.");
    foreach (EmsPolicy emsPolicy in emsPolicyList1)
    {
        if (emsPolicy.ExecutionContext != 1)
            emsPolicyList2.Add(emsPolicy);
        else
            Log.TraceInformation("[PowerShell] This is not AADJ/HAADJ device, skip user con
    }
    emsPolicyList1 = emsPolicyList2;
}
if (!flag1 && emsPolicyList1 != null && emsPolicyList1.Count > 0 && this.TimerIntern
{
    this.TimerInternal.Interval = 28800000.0;
    Log.TraceInformation($"[PowerShell] PS Script found, Timer is changed to {TimeSpan.
}
Log.TraceInformation("[PowerShell] After filter, get {0} policies for user {1} in ses
dictionary.Clear();
dictionary.Add("Step", (object) "PolicyProcess");
this.msg = string.Format((IFormatProvider) CultureInfo.InvariantCulture, "[PowerShell
dictionary.Add("EventMessage", (object) this.msg);
if (emsPolicyList1.Any<EmsPolicy>((Func<EmsPolicy, bool>) (policy => policy.PolicyTyp

```

**PowerShell = 8 Hour**  
**Required Apps = 1 hour**



# Asking the User to reboot (logoff/logon → Session Change)



```
protected override void OnSessionChange(SessionChangeDescription changeDescription)
{
    Log.TraceWarning("Session change detected.");
    Microsoft.Management.Services.IntuneWindowsAgent.AgentCommon.TelemetryManager.LogInfo(nameof (OnSessionChange), nameof (OnSessionCha
switch (changeDescription.Reason)
{
    case SessionChangeReason.SessionLogon:
        Log.TraceWarning("OnSessionChange: Logon");
        Thread.Sleep(30000);
        Log.TraceWarning("[PowerShell] Request PS Scripts for user logon.");
        Log.TraceWarning("[PowerShell] Check flighting to call OnTimer from OnSessionChange.");
        this.CallSelectedOnTimer((object) new CheckinReasonTypes(CheckinReason.UserLogOn), (ElapsedEventArgs) null);
        this.ProcessAppsOnSessionChange();
        Log.TraceWarning("[RebootHelper] CheckRebootFlagAndPop start...");
        RebootHelper.CheckRebootFlagAndPop(RebootCheckCaller.SessionLogon);
        Log.TraceWarning("[RebootHelper] CheckRebootFlagAndPop complete");
        HSLog.TraceWarning("[HS] Request health script check in for user logon.");
        PolicyPoller.Instance.Schedule((object) new CheckinReasonTypes(CheckinReason.UserLogOn), (ElapsedEventArgs) null);
        break;
}
```

It works.....

It will trigger the IME to kickoff the  
Win32 App and PowerShell Workload

Can MSFT make it better?



Fixing the  
Apps/Scripts  
Latency

# IC3!

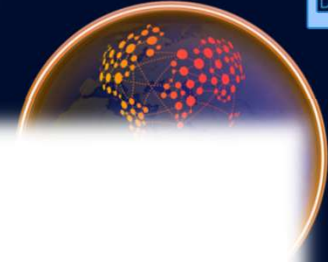
## This is where IC3 Comes in

Intelligent Conversation and Communications Cloud (I 3x3)



IC3 is the core infrastructure platform powering Microsoft Teams and Azure Communication Services (ACS) for instant communication

# IC3!



- Available
- Busy
- Do not disturb
- Be right back
- Appear away
- Appear offline

Teams and its “connection status” (Online/Offline/Away) is well known

```
using System;

#nullable disable
namespace Microsoft.Management.Services.IntuneWindowsAgent.IC3;

public class IC3ConnectionStats
{
    public int SuccessfulConnections { get; }
    public int FailedConnections { get; }
    public int ReconnectAttempts { get; }
    public int DisconnectAttempts { get; }

    public IC3ConnectionStats(
        int successfulConnections,
        int failedConnections,
        int reconnectAttempts,
        int disconnectAttempts)
    {
        this.SuccessfulConnections = successfulConnections;
        this.FailedConnections = failedConnections;
        this.ReconnectAttempts = reconnectAttempts;
        this.DisconnectAttempts = disconnectAttempts;
    }

    public override string ToString()
    {
        return string.Join(Environment.NewLine, "SuccessfulConnections=" + this.SuccessfulConnections.ToString(), "FailedConnections=" + this.FailedConnections.ToString(), "ReconnectAttempts=" + this.ReconnectAttempts.ToString(), "DisconnectAttempts=" + this.DisconnectAttempts.ToString());
    }
}
```

Home > Devices | Windows > Windows | Windows devices >

**-57767** ...

Search x << X Retire Wipe Delete Remote lock Sync

Overview

Manage

- Properties
- Monitor
- Device inventory
- Device query
- Hardware
- Discovered apps

Essentials

Device name : WVI

Management name : rudyooms\_Windows\_9/29/2025\_5:38 AM

Ownership : Corporate **Mockup**

Serial number : 2193-6

Phone number : ---

Device manufacturer : Microsoft Corporation

Online Status :

What if this is going to show up in Intune?



```
string s;  
try  
{  
    NotificationPayload notificationPayload = JsonSerializer.Deserialize<NotificationPayload>(json);  
    Log.TraceInformation("[IC3MessageListener] Extracted NotificationPayload with NotificationID: " + notificationPayload?.NotificationID);  
    string notificationId = notificationPayload?.NotificationID;  
    Log.TraceInformation($"[IC3MessageListener] NotificationID: {notificationId}, IC3version: {notificationPayload?.IC3Version}, timestamp: {no  
    try  
    {  
        await this.deviceActionCallback((object) json);  
    }  
}
```

Push request

WNS service  
(black box)

Notification

IC3 is a  
**replacement/improvement/addition**  
(or how you want to call it) for  
Windows Notification Services



# IC3!

```

Lock,
PinReset,
PinResetHybrid,
EPSignatureUpdate,
EPFullScan,
EPQuickScan,
Retire,
Wipe,
EnableLostMode,
DisableLostMode,
LocateDevice,
RebootNow,
MSCPinReset,
CleanPCRetainingUserData,
CleanPCWithoutRetainingUserData,
LogOutUser,
DeleteUser,
ShutDown,
WipePersistUserData,
None,
UpdateDeviceAccount,
AutomaticRedeployment,
RevokeAppleVppLicenses,
PlayLostModeSound,
Rename,
SetPinAction,
SaveDeviceLocation,
BypassActivationLock,
RequestRemoteAssistance,
SyncDevice,
SetDeviceName,
TimeLoop,
RotateBitLockerKeys,
RenameAndReboot,
RotateFileVaultKey,
CustomTextNotification,
ProtectedWipe,
RotateUserSubmittedPersonalRecoveryKey,
RequestDiagnostics,
ActivateDeviceEsim,
Deprovision,
Disable,
Reenable,
MoveDeviceToOrganizationalUnit,
RemoveDFCIManagement,
InitiateMDMKeyRecovery,
RemoteHelpLaunch,
OnDemandProactiveRemediation,
    
```

With it ALL ... ALL remote actions in Intune... (also the ones living in MDM world)

Can move over to the IME and could become instant!

**With it the IME becomes PUSH instead of PULL**

```

public enum SidecarNotificationActionType
{
    [EnumMember] None,
    [EnumMember] RemoteAssist,
    [EnumMember] Win32AppWorkload,
    [EnumMember] EndpointAnalytics,
    [EnumMember] RunRemediation,
    [EnumMember] IntunePivot,
    [EnumMember] ComplianceScripts,
    [EnumMember] WindowsRemoteHelpUnattended,
}
    
```



# IC3

When IC3 starts working.. The NotificationInfra.log will start showing cool stuff!!

```
[NotificationManager] Successfully retrieved Notification Infra client start flag from ECS: True. Has valid config: True N 4/19/2026 4:29:24 AM
[NotificationTelemetryService] Started telemetry reporting service N 4/19/2026 4:29:24 AM
[NotificationTelemetryService] Telemetry service started N 4/19/2026 4:29:24 AM
[NotificationClient] Initializing logger N 4/19/2026 4:29:24 AM
[NotificationClient] Logger initialized N 4/19/2026 4:29:24 AM
[NotificationClient] Successfully initialized IC3 client N 4/19/2026 4:29:24 AM
[NotificationClient] Configuration: ProviderId=https://login.windows.net, TroutorEndpoint=go-eu.trouter.communications.svc.cloud.microsoft N 4/19/2026 4:29:24 AM
[NotificationClient] Extracted 'exp' claim from JWT using JsonWebTokenHandler: 1776600194 N 4/19/2026 4:29:24 AM
[NotificationClient] Extracted token expiration from JWT: 04/19/2026 12:03:14 +00:00 N 4/19/2026 4:29:24 AM
[NotificationClient] Fetched new token, expires at: 04/19/2026 12:03:14 +00:00 with token: eyJ0eXAiOiJV1QjLCJub25jZSI6IjF1TTUuUzZ2ctamRLR2JTdVJmR2N6bUNYUk4tZ3Z5Z2c4RHFUc2t3TzgiLCJhbGciOiJSUzI1NiIsIng1dCI6IiUxc... N 4/19/2026 4:29:24 AM
[2026-04-19 11:29:24.835] Notification Infra Logger: Information: RegisterListener called N 4/19/2026 4:29:24 AM
[2026-04-19 11:29:24.851] Notification Infra Logger: Information: Start called N 4/19/2026 4:29:24 AM
[2026-04-19 11:29:24.871] Notification Infra Logger: Information: [C#v3] [1] [] []: Troutor client started with options: troutorHost=go-eu.trouter.communications.svc.cloud.microsoft, applicationName=intune, applicationVersion=3... N 4/19/2026 4:29:24 AM
[NotificationClient] Troutor client started successfully N 4/19/2026 4:29:24 AM
[2026-04-19 11:29:24.996] Notification Infra Logger: Information: [C#v3] [1] [] [WS#]: Creating new connection to wss://go-eu.trouter.communications.svc.cloud.microsoft:443/v3/c??timeout=55&epid=bbcc*****... N 4/19/2026 4:29:24 AM
[2026-04-19 11:29:25.006] Notification Infra Logger: Trace: [C#v3] [1] [] [WS#]: About to preconnect to wss://go-eu.trouter.communications.svc.cloud.microsoft:443/v3/c??timeout=55&epid=bbcc*****27eb N 4/19/2026 4:29:25 AM
[2026-04-19 11:29:25.008] Notification Infra Logger: Trace: [C#v3] [1] [] [WS#]: PreConnect returned wss://go-eu.trouter.communications.svc.cloud.microsoft:443/v3/c??timeout=55&epid=bbcc*****27eb N 4/19/2026 4:29:25 AM
[2026-04-19 11:29:25.010] Notification Infra Logger: Information: [C#v3] [1] [] [WS#1]: Starting to connect to wss://go-eu.trouter.communications.svc.cloud.microsoft:443/v3/c??timeout=55&epid=bbcc*****... N 4/19/2026 4:29:25 AM
[2026-04-19 11:29:25.010] Notification Infra Logger: Information: [C#v3] [1] [] [WS#1]: Connecting to wss://go-eu.trouter.communications.svc.cloud.microsoft:443/v3/c??timeout=55&epid=bbcc*****27eb N 4/19/2026 4:29:25 AM
[NotificationClient] Returning cached token, expires at: 4/19/2026 12:03:14 PM +00:00, token length: 1515 N 4/19/2026 4:29:25 AM
[2026-04-19 11:29:25.454] Notification Infra Logger: Trace: [C#v3] [1] [] [WS#1]: socket state Open sending 363 bytes N 4/19/2026 4:29:25 AM
[2026-04-19 11:29:25.454] Notification Infra Logger: Trace: [C#v3] [1] [LSN6E_QzrE-s34j4gx2KwQ] [WS#1]: Writing to socket N 4/19/2026 4:29:25 AM
[2026-04-19 11:29:25.454] Notification Infra Logger: Trace: [C#v3] [1] [] [WS#1]: socket state Open sending 363 bytes N 4/19/2026 4:29:25 AM
[2026-04-19 11:29:25.454] Notification Infra Logger: Information: [C#v3] [1] [LSN6E_QzrE-s34j4gx2KwQ] [WS#1]: OnMessageLoss: Events arguments do not contain any new dropped indicators N 4/19/2026 4:29:25 AM
[2026-04-19 11:29:25.501] Notification Infra Logger: Trace: [C#v3] [1] [] [WS#1]: Read message. Type: 'Text'; Read count: '8' eom: 'True'; N 4/19/2026 4:29:25 AM
[2026-04-19 11:29:25.538] Notification Infra Logger: Trace: [C#v3] [1] [] [WS#1]: Read message. Type: 'Text'; Read count: '8' eom: 'True'; N 4/19/2026 4:29:25 AM
[2026-04-19 11:30:08.258] Notification Infra Logger: Information: [C#v3] [1] []: SendRequest - got status code Accepted cv=UGoYGz6SJOSeCNfVSPYFyg.1 N 4/19/2026 4:30:08 AM
[2026-04-19 11:30:08.258] Notification Infra Logger: Trace: [C#v3] [1] []: Registration status is:True, was:False N 4/19/2026 4:30:08 AM
[NotificationClient] Connected to Troutor endpoint: https://pub-ent-pice-00-r.trouter.teams.microsoft.com:443/v3/f/LSN6E_QzrE-s34j4gx2KwQ/ N 4/19/2026 4:30:08 AM
[DEBUG_STATS][IC3Statistics] Incremented successful attempts from 0 to 1. ThreadId: 8, Call stack: IC3Statistics.IncrementSuccessfulConnection:0 -> <>c.<StartAsync>b__16_0:0 -> <<InvokeAllDelegates>g_InvokeSafe[0]_1>d'1... N 4/19/2026 4:30:08 AM
[2026-04-19 11:30:08.264] Notification Infra Logger: Information: [C#v3] [1] []: Successfully registered Troutor transport for the entire duration of the connection (6.16:37:36.9842405) N 4/19/2026 4:30:08 AM
```

Which is way better then the WNS event log in Windows... Which didn't show anything useful

# IC3



## In development for Microsoft Intune

```
public enum SidecarNotificationActionType
{
    [EnumMember] None,
    [EnumMember] RemoteAssist,
    [EnumMember] Win32AppWorkload,
    [EnumMember] EndpointAnalytics,
    [EnumMember] RunRemediation,
    [EnumMember] IntunePivot,
    [EnumMember] ComplianceScripts,
    [EnumMember] WindowsRemoteHelpUnattended,
}
```

### Remote Help connectivity updates for Windows devices

We're working to improve connectivity when using the Launch Remote Help feature in the Intune admin center for Windows devices. The improvement involves the addition of a new endpoint:

`*.trouter.communications.svc.cloud.microsoft.com`

For the best experience we recommend updating firewall rules to include the new endpoint once it becomes available.

For the current list of required network endpoints, see [Network requirements for PowerShell scripts and Win32 apps and Remote Help](#).

Applies to:

- ✓ Windows

- Microsoft.Management.Services.IntuneWindowsAgent.IC3
  - DefaultTrouterClientFactory
  - IC3Client
  - IC3ConnectionStats
  - IC3Constant
  - IC3OUTSIDE

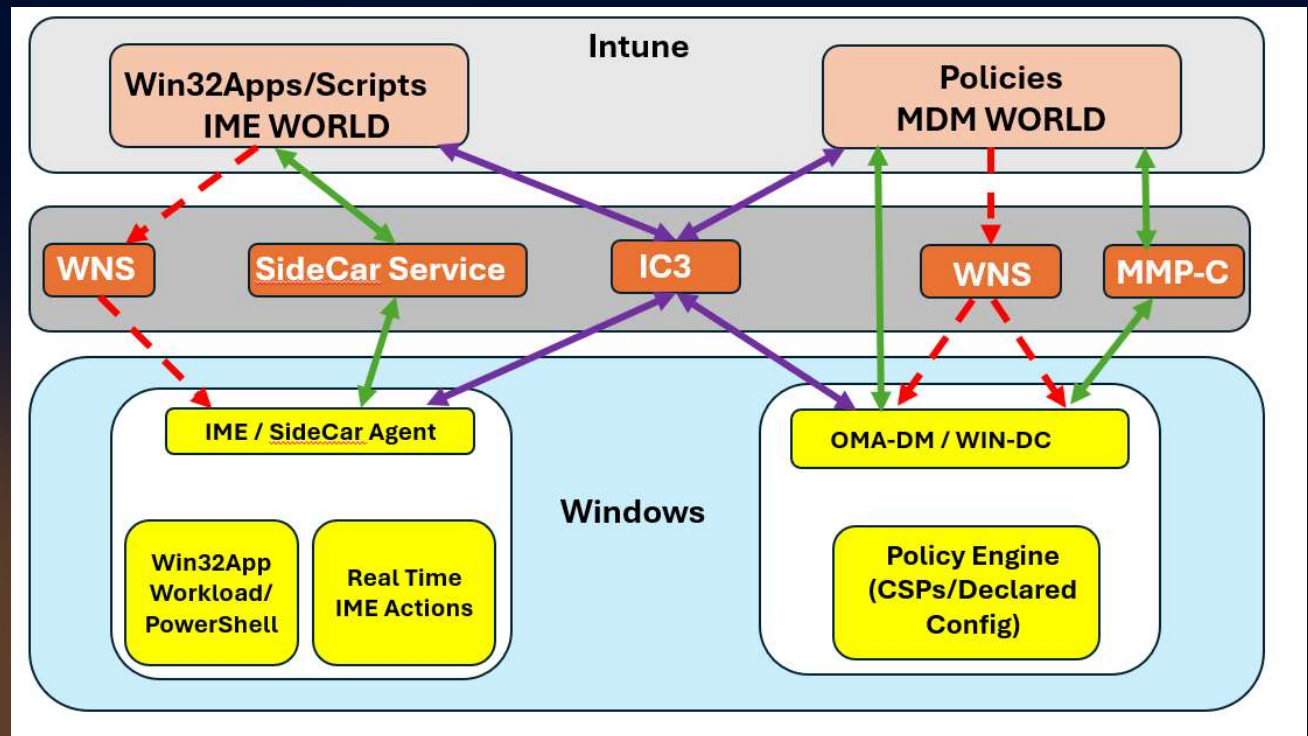
```
try
{
    Log.TraceInformation("Intune Start Trouter client");
    this._trouterClient?.Start();
}
catch (Exception ex)
{
    Log.Error("[IC3Client] Failed to start Trouter client: {0}", (object) ex);
    this.UpdateState(IC3Client.TrouterClientState.ConnectionError);
    throw;
}
```

Remote Help is the first one to start using IC3.  
Ensure you allow the new endpoint!!!

# Everything Starts Coming Together



Say goodbye to the IME Timers



# Summary



- A change in Intune will result in a WNS push to wake up the device to receive the policy
- WNS is a blackbox!
- **Intune today:** Policies apply on check-in, with lots of round trips. (get-set-get)
- Drift goes unnoticed until the next sync.
- **MMP-C fixes this** by enforcing the declared state locally.
- Devices use a **linked dual enrollment:** OMA-DM + MMP-C.
- One configuration document replaces dozens of SyncML transactions. (get-set)
- The IME is also evolving with IC3
- With IC3, Microsoft could ditch their dependency on WNS.

