



# **A Deep Dive into Location and Device-Based Access Control**

**Oktay Sari**

*Modern Workplace Consultant*

January 11, 2026

Version 1.0

## Document Information

Property	Value
Title	A Deep Dive into Location and Device-Based Access Control
Author	Oktay Sari, Modern Workplace Consultant
Company	All Things Cloud B.V.
Version	1.0
Date	January 11, 2026
Classification	Public
Publication	<a href="https://allthingscloud.blog">https://allthingscloud.blog</a>

## Version History

Version	Date	Author	Changes
1.0	January 11, 2026	Oktay Sari	Initial release

## Disclaimer

This document is provided for educational and informational purposes only. The techniques described should be thoroughly tested in a non-production environment before deployment.

The author and All Things Cloud B.V. are not responsible for any issues arising from the implementation of these techniques.

## Copyright Notice

© 2026 All Things Cloud B.V. All rights reserved. This document may be shared freely with attribution to the original author.

# Executive Summary

This whitepaper documents a Proof of Concept (POC) implementation for excluding specific users from Multi-Factor Authentication (MFA) while maintaining robust security through location-based and device-based access controls using Microsoft Entra ID Conditional Access policies.

## Key Takeaways

- ✓ 5-policy pattern: 2 Allow policies (with session controls) + 3 Block policies
- ✓ Device filters with explicit parentheses prevent unexpected evaluation results
- ✓ Guest/InPrivate browser mode bypasses device identity even on fully managed devices
- ✓ Cloud PC traffic requires special handling when using Microsoft Hosted Network
- ✓ Defense in depth: overlapping block policies catch edge cases

### Important Warning

- This whitepaper documents an MFA EXCLUSION pattern
- MFA should NOT be disabled without careful consideration
- 99.9% of compromised accounts do not have MFA enabled
- Use this pattern only when absolutely required by business constraints

# Table of Contents

Executive Summary .....	3
Table of Contents .....	4
1. A Word of Warning Before We Start .....	6
2. The Customer Requirement .....	7
The Requirements (In Plain English).....	7
3. The Challenge: Why This Isn't Straightforward .....	8
Challenge 1: Why Not Just Use "Require Compliant Device"? .....	8
Challenge 2: Why Session Controls Instead of Grant Controls? .....	8
Challenge 3: Cloud PC Traffic and Microsoft Hosted Network.....	9
Challenge 4: Device Filters and Boolean Logic .....	10
Challenge 5: Implicit Allow is Your Enemy .....	11
4. Design Summary.....	12
5. The Solution: 5 Conditional Access Policies.....	12
Policy Overview .....	12
Policy 1: CA001_Allow_Approved .....	13
Policy 2: CA002_Allow_CloudPC .....	13
Policy 3: CA003_Block_BYOD .....	14
Policy 4: CA004_Block_NonTrusted .....	14
Policy 5: CA005_Block_WrongProfile .....	15
6. Testing: The Test Matrix.....	16
Test Scenarios.....	16
7. Analyzing the Sign-in Logs: A Real-World Example .....	16
Exporting Sign-in Logs (optional).....	16
Key Fields to Analyze .....	17
Validating CA003: Finding the BYOD Gap.....	17
Validating CA005: Wrong Enrollment Profile .....	18
Deep Dive: Browser vs Native App Sign-ins (Same Device, Different Results!).....	19
What's Happening Here?.....	20
Why This Matters .....	21
Browser SSO Configuration and the Guest Mode Gotcha.....	22
Example: Cloud PC Traffic from Azure IP.....	22
8. Troubleshooting Guide.....	23
Check the Sign-in Logs.....	23

Policy → Root Cause → Resolution .....	23
Common Gotchas .....	23
9. Things You Could Do Differently.....	24
1. Use Azure Network Connection (ANC) for Windows 365.....	24
2. Global Secure Access .....	24
3. Use Compliant Device Requirement Instead of Enrollment Profile .....	25
4. Use Authentication Strengths Instead of Excluding MFA .....	25
5. Continuous Access Evaluation (CAE) Considerations .....	25
6. Consolidate to Fewer Policies.....	26
10. Device Filter Syntax Reference .....	26
Useful Operators .....	26
Behavior with Null Values .....	26
Security Considerations for Device Properties.....	27
FAQ.....	28
11. Conclusion .....	30
Resources .....	30
Conditional Access.....	30
Device Filters .....	30
Windows 365.....	30
Global Secure Access.....	31
Token Lifetime and CAE.....	31
Authentication.....	31
Browser SSO and Device Identity .....	31

"Because sometimes your security requirements make you question your life choices" 🤔

## 1. A Word of Warning Before We Start

Let me be crystal clear before we dive into this: **I do not advise excluding users from MFA.** Actually, I think it's... well, let's just say it's not the smartest move in the security playbook. MFA is one of the most effective security controls we have. According to Microsoft, [more than 99.9% of compromised accounts don't have MFA](#). Removing it is like taking the locks off your front door because you're tired of fumbling for your keys.

But here's the thing: sometimes customers come to you with requirements that make you tilt your head like a confused puppy. And as consultants, our job is to help them achieve their goals as securely as possible, even when those goals make us wanna cry out loud.

For the record: I tried talking them out of this. They insisted. This is me making sure they don't hurt themselves in the process. When the decision was made, my job shifted from "convince them otherwise" to "if we're doing this, let's do it as securely as humanly possible."

**This blog post documents a Proof of Concept (POC) implementation.** The techniques shown here should be thoroughly tested, validated, and approved by your security team before even thinking about production deployment. Consider this a learning exercise, not a production blueprint.

⚠️ This MFA exclusion approach is **not appropriate** for:

- **Regulatory/compliance environments** (HIPAA, PCI-DSS, SOX, GDPR often mandate MFA)
- **Privileged roles** (Global Admins, Security Admins, Finance, IT Admins)
- **External/guest users** (B2B scenarios should always require MFA)
- **Environments without robust monitoring** (if you can't detect anomalies, don't weaken controls)
- **Users with access to sensitive data** (HR, Legal, M&A, customer PII)

If any of these apply to your scenario, stop here and implement proper MFA instead. You should be doing that regardless, but I get it...And since you're still here, I'm guessing you've got one of those "it's complicated" situations.

With that disclaimer out of the way... Against my better judgment, let's continue...

## 2. The Customer Requirement

A customer approached me with a requirement that went something like this:

---

*"We have a specific group of users who need to access Microsoft 365 services without MFA, but ONLY under very specific conditions..."*

---

**Translation:** *"We want to poke holes in our security but make those holes as small as possible."*

After several cups of coffee and a few deep breaths, we defined the actual requirements:

### The Requirements (In Plain English)

1. **ONLY allow access without MFA** for users in **Security Group ABC** when:
  - User in Security Group ABC is on a trusted network location (specific IP: 123.123.123.123)
  - User is on an approved device (Entra ID Joined with specific enrollment profile OR Windows 365 Cloud PC)
2. **Always block BYOD devices** regardless of location (targeting the same security group)
3. **Allow M365 access from inside Cloud PC sessions.**

Here's the tricky part: once a user has legitimately connected to their Cloud PC from the trusted location, they need to access M365 services (SharePoint, Teams, Outlook, etc.) from within that Cloud PC. However, this traffic originates from Microsoft Azure infrastructure IPs, not your trusted corporate IP. So we need to explicitly allow this traffic to avoid blocking users from actually working inside their Cloud PC.

**Important clarification:** Users in Security Group ABC cannot connect to Windows 365 from "anywhere." They **MUST** be on the trusted location with an approved device to even sign-in using the Windows App and connect to their Cloud PC. The **CA002** policy only ensures they can use M365 apps once they're already inside the Cloud PC session.

Sounds simple enough, right?

### 3. The Challenge: Why This Isn't Straightforward

If you've worked with Conditional Access policies before, you might think: "Just create a policy that requires trusted location AND compliant device. Done!"

Well, there are some fun complications:

#### Challenge 1: Why Not Just Use "Require Compliant Device"?

The obvious solution would be to use the built-in Grant control "Require device to be marked as compliant" in combination with a trusted location requirement. But here's the catch:

**These are critical systems.** The customer explicitly stated that when a compliance check fails (maybe a policy hasn't synced yet, or there's a temporary issue), it should be:

- **Monitored** and reported
- **Investigated** and handled by IT
- **NOT an immediate block**

If we used the "Require compliant device" Grant control, any compliance failure would potentially lock users out of their critical applications. I know this depends on your compliance policy configuration. However, in this case, this is not acceptable.

Instead, we use **enrollment profile validation** through device filters. This approach:

- Validates the device was enrolled through an approved, controlled process (enrollment profile)
- Doesn't depend on real-time compliance state
- Provides a stable, predictable access control mechanism

Compliance is still monitored and enforced through separate Intune compliance policies and alerts, just not at the Conditional Access gate.

#### Challenge 2: Why Session Controls Instead of Grant Controls?

You might wonder: if we're not requiring compliance, why not just use other Grant controls?

Here's the problem: **Grant controls determine whether access is granted or blocked based on meeting a requirement.** Some of the available Grant options are:

- Block access
- Require multifactor authentication
- Require authentication strength
- Require device to be marked as compliant
- Require Microsoft Entra hybrid joined device
- etc.

None of these fit our scenario! We can't use:

- **Require MFA**; The whole point is to NOT require MFA
- **Authentication strength**; Unfortunately, also a no-go
- **Require compliant device**; critical systems can't tolerate immediate blocks (as explained above)
- **Require hybrid joined**; Not all devices are hybrid joined

Note on **Require Microsoft Entra hybrid joined device** from the docs:

- *Only supports domain-joined Windows down-level (before Windows 10) and Windows current (Windows 10+) devices.*
- *Doesn't consider Microsoft Edge in InPrivate mode as a Microsoft Entra hybrid joined device.*

More on the "Microsoft Edge" thingy later in this whitepaper...

So, what do we use when we want to **allow access without MFA** for specific scenarios? We use **Session controls** with a sign-in frequency. This creates an "Allow" policy that:

- Grants access to users who match the conditions
- Applies a session control (1-day sign-in frequency) instead of requiring MFA
- Works with our device filter to target specific approved devices

**Important:** Session controls don't block unauthorized access by themselves. They control re-authentication frequency for users who ARE granted access. That's why we need explicit Block policies (CA003, CA004, CA005) to catch everything else.

For more details on Grant vs Session controls, see [Microsoft's documentation on Grant controls](#) and [Session controls](#).

### Challenge 3: Cloud PC Traffic and Microsoft Hosted Network

This is where things get interesting. In this POC, the customer uses **Microsoft Hosted Network** for their Windows 365 Cloud PCs. This means:

- Microsoft manages the network infrastructure
- Cloud PC traffic egresses through Microsoft Azure's IP ranges
- You have NO control over the source IP addresses
- Traffic will NEVER come from your trusted corporate IP

**Why does this matter?** When a user inside their Cloud PC opens SharePoint or Teams, that request comes from an Azure IP address (e.g., 4.232.191.205), not your trusted location (123.123.123.123). If you simply block all non-trusted locations, you block your users from doing any actual work inside their Cloud PC!

## The Alternative: Azure Network Connection (ANC)

Now, here's something to think about for your own implementation. If you use **Azure Network Connection** instead of Microsoft Hosted Network, you could potentially:

- Route Cloud PC traffic through your own Azure VNet
- Egress traffic through your corporate network (via ExpressRoute or VPN)
- Have Cloud PC traffic appear to come from your trusted IP range
- Use standard location-based policies without the Cloud PC exclusion workaround

With ANC, you might be able to simplify this entire solution because all traffic (including from inside Cloud PCs) would come from your controlled network. This is worth exploring if you have the infrastructure and licensing to support it!

**However**, ANC comes with additional complexity:

- Requires Azure networking expertise
- Additional costs for VNet, gateways, and bandwidth
- More moving parts to manage and troubleshoot
- Potential latency considerations

For this POC, the customer used Microsoft Hosted Network, so we had to work around the Azure IP challenge. For more information, see [Windows 365 Network Requirements](#) and [Azure Network Connection](#).

## Challenge 4: Device Filters and Boolean Logic

Device filter syntax in Conditional Access can be... tricky. Unlike standard Boolean algebra where AND has higher precedence than OR, **Intune and Entra ID filters evaluate all operators with equal precedence, left to right.**

Microsoft explains this in their "[Back to school -- Using Boolean algebra correctly in complex filters](#)" article.

**The gotcha:** When your filter **ends with an OR condition**, the left-to-right evaluation can produce unexpected matches. Here's Microsoft's example:

**Intended:** Match (Dell Precision OR Latitude) AND (Prod Profile OR Lab Profile)

```
device.model -startsWith "Precision 3541" -or device.model -startsWith "Latitude" -and
device.enrollmentProfileName -eq "Autopilot Prod Profile" -or device.enrollmentProfileName
-eq "Autopilot Lab Profile"
```

**What left-to-right actually gives you:**

((("Precision 3541" OR "Latitude") AND "Prod Profile") OR "Lab Profile")

**Problem:** A Surface Laptop with "Lab Profile" would match!

**The trailing OR means ANY device with Lab Profile gets included.**

**An interesting workaround:** If you repeat a critical condition at the end of your filter, it acts as a "final gate":



```
device.model -contains "Cloud PC" -and device.trustType -eq "AzureAD" -or  
device.enrollmentProfileName -eq "profile1" -and device.trustType -eq "AzureAD"
```

This evaluates as ((model AND trustType) OR enrollmentProfile) AND trustType. Because trustType appears at both the beginning AND end, any match must be Entra Joined. This works, but it's fragile: if someone later modifies the filter or removes that trailing condition, it breaks silently.

**The recommended solution:** Use explicit parentheses. Yes, the filter builder can no longer display the query visually, but the clarity is worth it:

```
((device.enrollmentProfileName -eq "profile1") -and (device.trustType -eq "AzureAD")) -or  
((device.model -contains "Cloud PC") -and (device.trustType -eq "AzureAD"))
```


This ensures each condition group is properly evaluated before being OR'd together, and anyone reading the filter later will understand exactly what it does.


 Microsoft recommends using at least one system-defined or admin-configured device attribute when creating filter rules. [Learn more](#) 

Devices matching the rule:

- Include filtered devices in policy
- Exclude filtered devices from policy

You can use the rule builder or rule syntax text box to create or edit the filter rule.

And/Or	Property	Operator	Value	
	<Choose a property>	<Choose an ope...>	<Pick a property and operator first>	
+ Add expression				

Rule syntax ⓘ  Apply

```
((device.enrollmentProfileName -eq "Shared Device Profile") -and (device.trustType -eq "AzureAD")) -or ((device.model -contains "Cloud PC") -and (device.trustType -eq "AzureAD"))
```

Screenshot: Filter builder not showing the complex filter

## Challenge 5: Implicit Allow is Your Enemy

Conditional Access works on a "if no policy blocks, access is granted" principle. This means any scenario you don't explicitly handle could result in an implicit allow. Miss one edge case? Congratulations, you have a security gap.

That's why we ended up with 5 policies instead of 2 or 3. Every possible scenario needs to be explicitly handled.

## 4. Design Summary

Before diving into the implementation, here's the high-level design:

### Inputs:

- Trusted IP: 123.123.123.123 (corporate network)
- Approved devices: Entra ID Joined + specific enrollment profile (profile1) OR W365 Cloud PCs
- W365 constraint: Microsoft Hosted Network (Cloud PC traffic egresses from Azure IPs, not trusted IP)

**Output:** 5-policy pattern (2 Allow + Session, 3 Block)

### Decision flow:

- User on trusted IP + approved device → **CA001 allows** (1-day session)
- User inside Cloud PC accessing M365 → **CA002 allows** (Azure IPs, 1-day session)
- User on any device that's not Entra Joined → **CA003 blocks** (BYOD)
- User on non-trusted IP (except Cloud PC) → **CA004 blocks**
- User on trusted IP with wrong enrollment profile → **CA005 blocks**

## 5. The Solution: 5 Conditional Access Policies

After much whiteboarding, testing, and the occasional head-desk moment, we landed on a 5-policy solution. Why 5? Because troubleshooting matters, and when a user calls saying "I can't access SharePoint," you want the sign-in logs to tell you *exactly* which policy blocked them and why.

### Policy Overview

Policy	Type	Purpose
CA001_Allow_Approved	Allow + Session	Grant access to approved devices on trusted location
CA002_Allow_CloudPC	Allow + Session	Allow M365 access inside Cloud PC sessions (Azure IPs)
CA003_Block_BYOD	Block	Block all BYOD/unregistered devices everywhere
CA004_Block_NonTrusted	Block	Block access from non-trusted locations
CA005_Block_WrongProfile	Block	Block devices with wrong enrollment profile on trusted location

Let's break down each policy:

## Policy 1: CA001\_Allow\_Approved

**Purpose:** Grant access to users on approved devices at trusted locations

```
Name: CA001_Allow_Approved
Users: Security Group ABC
Cloud Apps: All cloud apps
Locations: Include: Trusted location (123.123.123.123)
Device Filter: Include filtered devices
Filter Syntax: ((device.enrollmentProfileName -eq "YourProfile") -and
(device.trustType -eq "AzureAD")) -or ((device.model -contains "Cloud PC") -and
(device.trustType -eq "AzureAD"))
Grant: Session sign-in frequency: 1 days
```

**What it does:** If you're on the trusted network AND using either an approved workstation (enrolled via specific profile) OR a Cloud PC, you get access with a 1-day session.

## Policy 2: CA002\_Allow\_CloudPC

**Purpose:** Allow M365 access from inside Cloud PC sessions

```
Name: CA002_Allow_CloudPC
Users: Security Group ABC
Cloud Apps: All cloud apps
Locations: All locations
Device Filter: Include filtered devices
Filter Syntax: ((device.model -contains "Cloud PC") -and (device.trustType -eq
"AzureAD"))
Grant: Session sign-in frequency: 1 days
```

**What it does:** When traffic comes from a Cloud PC device (identified by device properties, not location), allow access with a 1-day session.

**Why it's needed:** Remember, with Microsoft Hosted Network, Cloud PC traffic comes from Azure IPs. Without this policy, users would:

1. Successfully connect to their Cloud PC from the trusted location
2. Try to open SharePoint inside the Cloud PC BLOCKED (traffic comes from Azure IP)

This policy recognizes the Cloud PC device itself (which is Entra Joined and has "Cloud PC" in its model name) and allows the traffic regardless of source IP.

**Important:** This does NOT allow users to connect to their Cloud PC from anywhere. The Windows App connection itself is still subject to CA001/CA004. This policy only allows M365 traffic that originates from inside an already-established Cloud PC session.

## Policy 3: CA003\_Block\_BYOD

**Purpose:** Block all unmanaged devices, everywhere, always

```
Name: CA003_Block_BYOD
Users: Security Group ABC
Cloud Apps: All cloud apps
Locations: All locations
Device Filter: Include filtered devices
Filter Syntax: (device.trustType -ne "AzureAD")
Grant: Block
```

**What it does:** Any device that is NOT Entra ID Joined gets blocked. This catches:

- Personal devices (BYOD)
- Entra ID Registered devices (not the same as Joined!)
- Any unmanaged device

**The magic of \-ne\ with null values:** BYOD devices have null for trustType. The expression `device.trustType -ne "AzureAD"` evaluates to TRUE for null values, so BYOD devices get caught by this filter. See [Microsoft's documentation on device filters](#) for more details on how null values are evaluated.

## Policy 4: CA004\_Block\_NonTrusted

**Purpose:** Block access from non-trusted locations (except Cloud PC sessions)

```
Name: CA004_Block_NonTrusted
Users: Security Group ABC
Cloud Apps: All cloud apps
Locations: Include: Any location, Exclude: Trusted location (123.123.123.123)
Device Filter: Exclude filtered devices
Filter Syntax: ((device.model -contains "Cloud PC") -and (device.trustType -eq "AzureAD"))
Grant: Block
```

**What it does:** Block access from any location that isn't your trusted IP, EXCEPT for Cloud PC devices.

**Why exclude Cloud PC:** If we didn't exclude Cloud PCs, the M365 traffic from inside Cloud PC sessions would be blocked (since it comes from Azure IPs, not the trusted location). The device filter exclusion recognizes legitimate Cloud PC traffic and lets it through while blocking everything else from non-trusted locations.

## Policy 5: CA005\_Block\_WrongProfile

**Purpose:** Block devices with incorrect enrollment profile on trusted location

```
Name: CA005_Block_WrongProfile  
Users: Security Group ABC  
Cloud Apps: All cloud apps  
Locations: Include: Trusted location (123.123.123.123)  
Device Filter: Exclude filtered devices  
Filter Syntax: ((device.enrollmentProfileName -eq "YourProfile") -and  
(device.trustType -eq "AzureAD")) -or ((device.model -contains "Cloud PC") -and  
(device.trustType -eq "AzureAD"))  
Grant: Block
```

**What it does:** On the trusted location, block any Entra Joined device that doesn't have the correct enrollment profile.

**Why it's needed:** Without this, someone could bring an Entra Joined device with a different enrollment profile and get access. The exclude filter means only correctly enrolled devices and Cloud PCs are excluded from the block.

### Pro tip

Before relying on **model** filters in production, validate the actual model strings that exist in your tenant. Query your Entra ID device inventory to see what Cloud PCs and other devices report as their model values. This helps you write accurate filters and avoid surprises from unexpected model string formats.


**Warning:** When users switch to a new Cloud PC model, policies that rely on the model filter might not catch them if these policies are not correctly formatted!

## 6. Testing: The Test Matrix

Before deploying anything to production (you ARE testing in a POC environment first, right?), we created a test matrix. Here's what we tested:

### Test Scenarios

#	Scenario	Device Type	Enrollment Profile	Location	Expected	Actual	Policy
1	BYOD + Trusted	Personal	None	Trusted	Block	Block	CA003
2	BYOD + Non-Trusted	Personal	None	Non-Trusted	Block	Block	CA003+CA004
3	Approved + Trusted	Workstation	YourProfile	Trusted	Allow	Allow	CA001
4	Cloud PC + Azure IP	Cloud PC	W365 Profile	Azure IP	Allow	Allow	CA002
5	Approved Device + Non-Trusted	Workstation	YourProfile	Non-Trusted	Block	Block	CA004
6	Wrong Profile + Trusted	macOS	DifferentProfile	Trusted	Block	Block	CA005
7	Wrong Profile + Non-Trusted	macOS	DifferentProfile	Non-Trusted	Block	Block	CA004
8	Windows App from Non-Trusted	Any	Any	Non-Trusted	Block	Block	CA004
9	M365 inside Cloud PC	Cloud PC	W365 Profile	Azure IP	Allow	Allow	CA002

All tests passed! 

## 7. Analyzing the Sign-in Logs: A Real-World Example

Let me walk you through how we validated the policies were working by analyzing sign-in logs.

### Exporting Sign-in Logs (optional)

4. Navigate to **Entra ID** → **Monitoring** → **Sign-in logs**
5. Filter by user or time range
6. Click **Download** → **Download CSV**

You'll get files like:

- InteractiveSignIns\_2026-01-09.csv
- NonInteractiveSignIns\_2026-01-09.csv

## Key Fields to Analyze

Field	What to Look For
Status	Success / Failure / Interrupted
IP address	Is it your trusted IP? Or Azure IP (Cloud PC)?
Device ID	null = BYOD, GUID = registered device
Join Type	"Azure AD joined" / "Azure AD registered" / empty
Compliant	True / False / empty
Conditional Access	Which policy was applied
Failure reason	Why was access blocked

## Validating CA003: Finding the BYOD Gap

To prove that CA003\_Block\_BYOD was essential, we temporarily disabled it and tested BYOD access from the trusted location. Here's what happened:

### Without CA003 (testing the gap):

```
Request ID: c0551ea4-44d5-4b2b-979c-XXXXXXXXXXXXX
Status: Success (!)
IP address: 123.123.123.123 (Trusted)
Device ID: null
Join Type: (empty)
Operating System: Windows 10
Application: OfficeHome
```

This confirmed our concern: without an explicit BYOD block policy, personal devices from the trusted location got through because no policy explicitly blocked them. The implicit allow kicked in!

### With CA003 enabled:

```
Request ID: 54ae5a8b-36bc-40f7-ba94-XXXXXXXXXXXXX
Status: Failure
IP address: 123.123.123.123 (Trusted)
Device ID: null
Join Type: (empty)
Operating System: macOS
Application: Microsoft Teams Web Client
Failure reason: Access has been blocked by Conditional Access policies.
Blocking Policy: CA003_Block_BYOD
```

CA003 caught it! BYOD devices are now blocked even from the trusted location.

## Activity Details: Sign-ins

Basic info	Location	Device info	Authentication Details	<b>Conditional Access</b>	Report-only
<input type="text" value="Search"/>					
Policy Name ↑↓	Grant Controls ↑↓	Session Controls ↑↓	Result ↑↓		
CA0003: _Block_BYOD	Block		Failure		

Screenshot: Activity Details: Sign-ins CA0003

## Validating CA005: Wrong Enrollment Profile

Similarly, we tested CA005 by temporarily disabling it and using a macOS device enrolled with a different profile (not "YourProfile"):

**Without CA005 (testing the gap):**

```
Request ID: f660f3e4-9b00-4fbf-acc2-XXXXXXXXXXXXX
Status: Success (!)
IP address: 123.123.123.123 (Trusted)
Device ID: 573a91ff-d641-4d87-XXXX-XXXXXXXXXXXXX
Join Type: Azure AD joined
Compliant: True
Managed: True
Enrollment Profile: DifferentProfile
```

This device was Entra Joined and compliant but had the wrong enrollment profile. Without CA005, it got through!

**With CA005 enabled:**

```
Request ID: b92bca5c-dd72-4e8a-8612-XXXXXXXXXXXXX
Status: Failure
IP address: 123.123.123.123 (Trusted)
Device ID: 573a91ff-d641-4d87-XXXX-XXXXXXXXXXXXX
Join Type: Azure AD joined
Failure reason: Access has been blocked by Conditional Access policies.
Blocking Policy: CA005_Block_WrongProfile
```

CA005 caught it! The device had a different enrollment profile and was correctly blocked.

## Deep Dive: Browser vs Native App Sign-ins (Same Device, Different Results!)

This is where things get interesting. During our testing, we observed something that's critical to understand: **the same device can trigger different policies depending on how the user signs in.**

Let me show you two sign-in attempts from the same user, on the same device, just 21 seconds apart:

### \*\*Sign-in 1: Browser in Guest Mode (Microsoft Teams Web Client)\*\*

Policy	Result	Device Filter
CA003_Block_BYOD	❌ Failure	"Device filter rule included"
CA005_Block_WrongProfile	❌ Failure	"Device filter rule excluded"

**Request ID:** 54ae5a8b-36bc-40f7-ba94-XXXXXXXXXXXX

**Time:** 1/9/2026, 1:13:54 PM

**Application:** Microsoft Teams Web Client (Browser - Edge in Guest Mode)

**Device ID:** (empty/null)

**Join Type:** (empty)

**Compliant:** (empty)

**IP:** 123.123.123.123 (Trusted)

**Status:** Failure

## Activity Details: Sign-ins

Basic info   Location   Device info   Authentication Details   **Conditional Access**   Report-only

Search			
Policy Name ↑↓	Grant Controls ↑↓	Session Controls ↑↓	Result ↑↓
CA0003: _Block_BYOD	Block		Failure
CA0005: _Block_WrongPro...	Block		Failure

Screenshot: Activity Details: Sign-ins CA0003 & CA0005

**Important context:** This macOS device has:

- Company Portal installed
- Device enrolled in Intune
- Platform SSO deployed and configured (which includes the SSO app extension)
- Entra ID joined
- Compliant status

Everything is properly configured for browser SSO to work! But the user opened the browser in Guest mode ("Browse as Guest").

## Policies triggered:

Wait, both CA003 AND CA005 blocked this? But if CA003 (BYOD block) fired, shouldn't this be a BYOD device? And the device IS enrolled with Platform SSO! Let's look at the next sign-in...

## \*\*Sign-in 2: Native App (Windows App for macOS)\*\*

Policy	Result	Why
CA003_Block_BYOD	✔ Not Applied	trustType = AzureAD, so filter doesn't match
CA005_Block_WrongProfile	✘ Failure	Device is Entra Joined but has wrong enrollment profile

```
Request ID: b92bca5c-dd72-4e8a-8612-XXXXXXXXXXXX
Time: 1/9/2026, 1:14:15 PM
Application: Windows App - macOS (Native app)
Device ID: 573a91ff-d641-4d87-XXXX-XXXXXXXXXXXX
Join Type: Azure AD joined
Compliant: true
Managed: true
IP: 123.123.123.123 (Trusted)
Status: Failure
```

CA0001:  _Allow_Approved	Sign-in frequency	Not Applied
CA0002:  _Allow_CloudPC	Sign-in frequency	Not Applied
CA0003:  _Block_BYOD	Block	Not Applied
CA0004:  _Block_NonTrusted	Block	Not Applied
CA0005:  _Block_WrongPro...	Block	Failure

Screenshot: Activity Details: Sign-ins CA0005

## What's Happening Here?

**Same device. Same user. Same location. Completely different policy evaluation!**

The key difference is **how the application authenticates with Entra ID**, and in the browser case, **Guest mode completely bypasses all device identity mechanisms**.

### Browser sign-in (Edge in Guest Mode):

- **Guest mode has no browser profile** → No access to stored credentials or certificates
- **No SSO extension context** → Even though Platform SSO is deployed (which includes the SSO app extension), Guest mode can't access it
- **No PRT access** → The Primary Refresh Token is tied to the user profile, not available in Guest mode

- **(macOS) No device certificate** → Can't be presented without profile context to access the Keychain
- Entra sees the device as "Unknown" with trustType = null
- CA003's filter (device.trustType -ne "AzureAD") evaluates to TRUE (because null -ne "AzureAD" is TRUE)
- CA005's exclude filter also doesn't match (because null -eq "AzureAD" is FALSE)
- Result: Both policies block!

#### **Native app sign-in (Windows App):**

- The Windows App properly uses the device's PRT for authentication
- It has full access to the Keychain and device certificates
- Entra sees the full device identity: Azure AD joined, compliant, managed
- CA003's filter (device.trustType -ne "AzureAD") evaluates to FALSE (because "AzureAD" -ne "AzureAD" is FALSE)
- CA003 does NOT apply (device is not included in the filter)
- CA005's exclude filter checks the enrollment profile, and it doesn't match "profile1"
- Result: Only CA005 blocks (the correct behavior!)

## **Why This Matters**

This example demonstrates several critical points about browser-based authentication and device identity:

### **1. Guest/InPrivate mode completely bypasses device identity.**

**Documented behavior (Microsoft):** Private browsing modes break device identity checks.

*"InPrivate session is not supported in Conditional Access for all browsers, as there's no concept of a signed in profile in this mode and hence browser fails to retrieve PRT and device claims when performing authentication." --- [Microsoft Q&A: Hybrid Azure AD joined devices show up as Unknown](#)*

**Observed in this POC:** Edge Guest mode resulted in null device ID and empty trustType in sign-in logs, even on a fully managed, Entra Joined, compliant device with Platform SSO deployed.

### **2. Device identity requires proper browser configuration.**

**Documented behavior (Microsoft):** Browsers need profile access to pass device identity.

*"Microsoft Edge 85+ requires the user to be signed in to the browser to properly pass device identity. Otherwise, it behaves like Chrome without the Microsoft Single Sign On extension." --- [Conditional Access Conditions](#)*

### **3. (macOS only) Platform SSO includes the SSO app extension, but Guest mode bypasses both.**

In this example we used a macOS device with Platform SSO enabled (PSSO). PSSO is an enhancement that includes the SSO app extension functionality. You don't deploy them separately. Guest mode bypasses the entire mechanism because it can't access the browser profile where the SSO extension operates. See the "Browser SSO Configuration" section below for deployment options.

### **4. Defense in depth still works!**

**Observed in this POC:** Even though CA003 "incorrectly" fired (the device isn't actually BYOD, it just appears that way), the access was still blocked. The user can't bypass security just by using Guest mode. This is actually a security WIN: your block policies catch unknown devices regardless of why they're unknown.

#### 5. CA005 is your safety net for native apps.

**Observed in this POC:** When device identity IS properly passed (native apps, browsers with proper profile), CA005 correctly identifies that this Entra Joined device has the wrong enrollment profile and blocks it.

#### 6. Sign-in logs tell the full story.

By comparing these two requests, we can understand exactly why access was blocked and whether it was the intended policy or a side effect of missing device identity. Always check:

- Device ID (empty = no device identity passed)
- Join Type (empty = unknown device)
- Client app type (Browser vs Mobile apps and desktop clients)

### Browser SSO Configuration and the Guest Mode Gotcha

**The key takeaway:** Guest mode, InPrivate, and Incognito create sandboxed browser sessions that cannot access the Keychain, device certificates, SSO extension, or PRT. No matter how perfectly your device is configured, private browsing bypasses it all.

### Example: Cloud PC Traffic from Azure IP

Here's what legitimate Cloud PC traffic looks like:

```
Request ID: d3ce2624-0231-4626-8d2e-XXXXXXXXXXXXX
Status: Success
IP address: 4.232.191.205 (Azure IP - NOT trusted location!)
Device ID: 383d507a-89ac-4763-XXXX-XXXXXXXXXXXXX
Join Type: Azure AD joined
Compliant: True
Managed: True
Operating System: Windows 10
Application: Office365 Shell WCSS-Client
Applied Policy: CA002_Allow_CloudPC
```

Notice the IP is an Azure IP, not the trusted location. Without CA002, this would have been blocked!

## 8. Troubleshooting Guide

When users call saying "I can't access \[insert app here\]," here's how to diagnose:

### Check the Sign-in Logs

7. Go to **Entra ID** → **Sign-in logs**
8. Find the user's failed sign-in
9. Click on it → **Conditional Access** tab
10. Look for which policy shows "Failure"

### Policy → Root Cause → Resolution

Blocking Policy	Root Cause	Resolution
CA003_Block_BYOD	Device is not Entra ID Joined	Use a corporate managed device
CA004_Block_NonTrusted	User is outside trusted network	Connect from trusted location or use Cloud PC
CA005_Block_WrongProfile	Device has wrong enrollment profile	Re-enroll device with correct Intune profile

### Common Gotchas

#### "But I'm on a corporate device!"

- Check if it's Entra ID **Joined** (not just Registered)
- Verify the enrollment profile name matches what's in the policy

#### "I'm on the VPN but still blocked!"

- VPN might not route traffic through the trusted IP
- Check the actual source IP in the sign-in logs

#### "It worked yesterday!"

- Check if any policies were recently changed
- Token caching can cause delays: existing sessions might still work until the access token expires (default 60-90 minutes)
- Use InPrivate/Incognito mode to force a fresh authentication and test current policy state
- If using [Continuous Access Evaluation \(CAE\)](#), tokens can be valid for up to 28 hours, but critical events are still enforced in near real-time

#### "I can connect to my Cloud PC but can't open SharePoint inside it!"

- Check if CA002 is properly configured
- Verify the Cloud PC device filter is correct
- Look at the sign-in logs for the Azure IP traffic

## 9. Things You Could Do Differently

This solution works for this specific use case, but here are some alternatives to consider:

### 1. Use Azure Network Connection (ANC) for Windows 365

As mentioned earlier, if you use ANC instead of Microsoft Hosted Network:

- Cloud PC traffic can egress through your corporate network
- All traffic appears to come from your trusted IP range
- You could potentially eliminate CA002 entirely
- Standard location policies would work without Cloud PC exclusions

#### Trade-offs:

- Additional Azure infrastructure required
- More complex setup and maintenance
- Potential latency impact
- Additional costs

This is worth serious consideration if you're planning a larger Windows 365 deployment! See [Azure Network Connection documentation](#) for more details.

### 2. Global Secure Access

Microsoft's **Global Secure Access** (part of Microsoft Entra Suite) offers another elegant solution to the Cloud PC IP problem.

With Global Secure Access, you can:

- Define a "Compliant Network" check in Conditional Access
- Traffic is validated through the Global Secure Access client
- Works regardless of source IP address
- Provides consistent policy enforcement for all traffic

#### How it helps this scenario:

Instead of excluding Cloud PCs from location policies and trusting the device filter, you could use the "Compliant Network" condition. Users would need the Global Secure Access client running, which validates they're connecting through the proper channels.

From [Microsoft's documentation](#):

"This compliant network check ensures users connect via the Global Secure Access service for their specific tenant and are compliant with security policies enforced by administrators."

#### Key benefits:

- Removes the need to maintain IP address lists
- Works regardless of source IP (solves the Cloud PC Azure IP problem)
- Provides defense against token theft/replay attacks
- Integrates with Continuous Access Evaluation

### Requirements:

- Microsoft Entra Suite or standalone Global Secure Access license
- Global Secure Access Client deployed to devices
- Additional configuration and infrastructure

If you're already investing in Microsoft Entra Suite, Global Secure Access is worth exploring! See the [Global Secure Access documentation](#) for more information.

## 3. Use Compliant Device Requirement Instead of Enrollment Profile

If your organization CAN tolerate immediate blocks for non-compliant devices, you could simplify the device filter to:

```
device.isCompliant -eq True
```

**Pros:** Simpler filter, no need-to-know exact profile names, validates ongoing device health

**Cons:** Less granular control, compliance sync delays can cause unexpected blocks, not suitable for mission-critical systems that need high availability

## 4. Use Authentication Strengths Instead of Excluding MFA

Instead of excluding MFA entirely, consider:

- Requiring phishing-resistant authentication (FIDO2 keys)
- Using Windows Hello for Business
- Certificate-based authentication

**This is what I'd recommend!** You get strong authentication without traditional MFA prompts. The user experience is seamless (biometric or PIN), but you maintain security. See [Authentication Strengths documentation](#) for configuration details.

## 5. Continuous Access Evaluation (CAE) Considerations

If you implement this solution, be aware of Continuous Access Evaluation and its impact on token lifetime and policy enforcement.

From [Microsoft's CAE documentation](#):

*"Because risk and policy are evaluated in real time, some resource APIs token lifetime can increase by up to 28 hours. These long-lived tokens are proactively refreshed by the Microsoft Authentication Library (MSAL), increasing the resiliency of your applications."*

### Key CAE behaviors:

- Default token lifetime without CAE: 60-90 minutes
- Token lifetime with CAE: up to 28 hours
- Critical events (password change, account disabled, session revoked) are enforced in near real-time
- Location changes for CAE-aware apps trigger re-evaluation

### What this means for your rollout:

- Policy changes may take up to 28 hours to affect existing CAE sessions
- For immediate enforcement, consider disabling/re-enabling user accounts or forcing password changes
- Simple "Revoke Sessions" may not be immediately effective due to extended token lifetime
- CAE provides better security overall through real-time event evaluation

For detailed implementation guidance, see [Secure applications with Continuous Access Evaluation](#).

## 6. Consolidate to Fewer Policies

You could potentially reduce to 3-4 policies by using more complex filters. However:

- Troubleshooting becomes harder
- Sign-in logs are less clear about why something was blocked
- Maintenance becomes more error-prone

I prefer clarity over cleverness in security policies.

## 10. Device Filter Syntax Reference

For those who want to dive deeper into the filter syntax:

### Useful Operators

Operator	Description	Example
-eq	Equals	device.trustType -eq "AzureAD"
-ne	Not equals	device.trustType -ne "AzureAD"
-contains	Contains string	device.model -contains "Cloud PC"
-startsWith	Starts with	device.displayName -startsWith "WKS-"
-and	Logical AND	(condition1) -and (condition2)
-or	Logical OR	(condition1) -or (condition2)

### Behavior with Null Values

Expression	BYOD Result (null)	Why
device.trustType -eq "AzureAD"	FALSE	null ≠ "AzureAD"
device.trustType -ne "AzureAD"	TRUE	null is not equal to "AzureAD"
device.isCompliant -eq True	FALSE	null ≠ True
device.isCompliant -ne True	TRUE	null is not equal to True

This is crucial for BYOD detection:

**Pro tip:** Use -ne operators to catch BYOD devices!

## Security Considerations for Device Properties

Property	Controlled By	Can Be Modified By User?	Security Notes
enrollmentProfileName	Intune/System	No	Set during enrollment; requires re-enrollment to change. Controlled by Autopilot, ADE, or Android enrollment profiles.
trustType	Microsoft Entra/System	No	System-controlled. Values: "AzureAD" (Joined), "ServerAD" (Hybrid), "Workplace" (Registered). Cannot be spoofed without re-registering the device.
isCompliant	Intune/System	No	Determined by Intune compliance policies. Requires device to pass all assigned compliance checks.
model	Hardware/Registry	Potentially	Sourced from hardware/registry. Microsoft notes it "could be spoofed by a person with access to the device."
displayName	User	Yes	Users can rename devices. Never rely on this for security decisions!
manufacturer	Hardware/Registry	Potentially	Similar to model, sourced from registry entries.

Understanding which properties can be trusted is critical for security:

### Important security note from Microsoft:

*"Customers should avoid using Entra device properties (ones that can be modified or manipulated by an end user) just on its own when creating a device filter rule. Microsoft recommends using these properties in conjunction (use of AND clause) with some of the other properties on the device that are not modifiable by the end user."*

In our solution, we always combine enrollmentProfileName or model with trustType -eq "AzureAD" to ensure the device is legitimately Entra Joined before trusting other properties.

**Pro tip:** Before relying on model filters in production, validate the actual model strings that exist in your tenant. Query your Entra ID device inventory to see what Cloud PCs and other devices report as their model values. This helps you write accurate filters and avoid surprises from unexpected model string formats.

Also note from Microsoft's [enrollment restrictions documentation](#):

"Enrollment restrictions are not security features. Compromised devices can misrepresent their character. These restrictions are a best-effort barrier for non-malicious users."

While trustType, enrollmentProfileName, and isCompliant are system-controlled and much harder to spoof than user-modifiable properties, they should be considered part of a defense-in-depth strategy, not the sole security control.

## FAQ

### **Q: Why 5 policies instead of combining them?**

**A:** Troubleshooting. When a user calls, the sign-in logs will show exactly which policy blocked them. "CA003\_Block\_BYOD" tells you immediately it's a BYOD issue. A generic "CA001\_Everything" policy would require digging into device details.

### **Q: What happens if a user is in multiple groups with conflicting policies?**

**A:** Conditional Access follows "most restrictive wins." If any policy blocks, access is blocked. If multiple policies allow with different controls, all controls apply. See [Building Conditional Access policies](#) for details.

### **Q: How do I test this without affecting production users?**

**A:**

11. Create a test security group
12. Enable policies in "Report-only" mode first
13. Analyze the "What If" tool
14. Test with specific users before broader rollout

See [Plan your Conditional Access deployment](#) for comprehensive testing guidance.

### **Q: The Cloud PC exclusion seems risky. Can someone spoof a Cloud PC?**

**A:** The filter checks both `device.model -contains "Cloud PC"` AND `device.trustType -eq "AzureAD"`. To spoof this, an attacker would need an Entra ID Joined device with "Cloud PC" in its model string. Cloud PC model names are controlled by Microsoft during provisioning, making this difficult to spoof. That said, always validate the actual model strings in your tenant before relying on this filter. If you want even more security, consider Global Secure Access or Azure Network Connection as alternatives. See the "Security Considerations for Device Properties" section above.

### **Q: What about Entra ID Registered devices vs. Joined?**

**A:** They're different!

- **Registered (trustType: "Workplace"):** Personal device that's been registered (BYOD scenario)
- **Joined (trustType: "AzureAD"):** Device is organization-owned and domain-joined to Entra ID
- **Hybrid Joined (trustType: "ServerAD"):** Device is joined to both on-premises AD and Entra ID

Our CA003 policy blocks Registered devices because their trustType is "Workplace" not "AzureAD".

### **Q: What about token lifetime and Continuous Access Evaluation?**

**A:** Access tokens have a default lifetime of 60-90 minutes. With CAE enabled, tokens can be valid for up to 28 hours, but critical events are still enforced in near real-time. If someone authenticated before a policy change, they might retain access until their token expires or a critical event triggers re-evaluation. For immediate enforcement, consider user account disable/enable cycles or password resets. See [CAE documentation](#) for complete details.

**Q: My organization doesn't use Windows 365. Can I simplify this?**

**A:** Absolutely! Remove the Cloud PC portions from all filters:

- Remove CA002 entirely
- Simplify CA001 filter to just your enrollment profile
- Remove Cloud PC exclusion from CA004 and CA005

**Q: Is this approach Microsoft-recommended?**

**A:** Microsoft recommends using Allow + Block policy combinations rather than relying on implicit deny. This solution follows that pattern. However, Microsoft's primary recommendation is to NOT exclude users from MFA. Use this approach only when absolutely necessary.

**Q: What's the difference between Microsoft Hosted Network and Azure Network Connection?**

**A:**

- **Microsoft Hosted Network:** Microsoft manages everything. Cloud PC traffic egresses through Microsoft's Azure IPs. Simple to set up, but you have no control over network paths.
- **Azure Network Connection (ANC):** You connect Windows 365 to your own Azure VNet. Traffic can be routed through your corporate network. More complex, but gives you control over egress IPs.

For this POC, Microsoft Hosted Network was used, which is why we needed the CA002 workaround for Cloud PC traffic.

**Q: Why can't I just use Grant: "Require compliant device" for this scenario?**

**A:** Because in this case, these are critical systems. The customer explicitly required that compliance failures should be monitored, reported, and investigated, but NOT (immediately) block access. Using "Require compliant device" would potentially lock users out the moment any compliance check fails, even temporarily. Instead, we use enrollment profile validation for access control and handle compliance through separate monitoring and alerting.

**Q: Why does my managed device show up as "Unknown" when I use Guest mode or InPrivate browsing?**

**A:** Guest mode, InPrivate (Edge), Incognito (Chrome), and Private Browsing (Safari/Firefox) all create sandboxed browser sessions that cannot access your device's credentials or the SSO extension. Microsoft explicitly documents this:

*"InPrivate session is not supported in Conditional Access for all browsers, as there's no concept of a signed in profile in this mode and hence browser fails to retrieve PRT and device claims when performing authentication."*

Even if your device is:

- Entra ID Joined
- Enrolled in Intune
- Has Platform SSO deployed (macOS)
- Is fully compliant

...Guest mode bypasses ALL of this because it can't access the Keychain, device certificates, or SSO extension. The browser has no way to prove the device identity to Entra ID.

**Bottom line:** Don't use Guest mode or private browsing when accessing corporate resources that require device-based Conditional Access. Use a regular browser window signed into your browser profile.

## 11. Conclusion

So there you have it: a 5-policy solution to exclude specific users from MFA while maintaining as much security as possible. Is it ideal? No. Is it sometimes necessary? Unfortunately, yes.

Remember:

- **Test extensively** before production (Report-only mode is your friend!)
- **Document everything** (you'll thank yourself later)
- **Monitor sign-in logs** regularly for anomalies
- **Consider alternatives** like authentication strengths, Global Secure Access, or Azure Network Connection
- **Understand token lifetime** and CAE when planning rollouts
- **Keep pushing** for proper MFA implementation!

And please, for the love of all things secure, treat this as a stepping stone, not a final destination. The goal should always be to eventually bring these users under proper MFA protection.

Stay secure out there! 🙌

## Resources

### Conditional Access

- [Conditional Access Overview](#)
- [Grant Controls](#)
- [Session Controls](#)
- [Plan Your Conditional Access Deployment](#)
- [Building Conditional Access Policies](#)
- [How MFA Works](#)

### Device Filters

- [Filter for Devices Condition](#)
- [Back to School: Boolean Algebra in Complex Filters](#)
- [Assignment Filter Properties Reference \(Intune\)](#)

## Windows 365

- [Windows 365 Network Requirements](#)
- [Azure Network Connection](#)
- [Windows 365 and Conditional Access](#)

### **Global Secure Access**

- [What is Global Secure Access?](#)
- [Enable Compliant Network Check](#)
- [Universal Conditional Access](#)

### **Token Lifetime and CAE**

- [Continuous Access Evaluation](#)
- [Secure Applications with CAE](#)
- [Session Lifetime Policies](#)

### **Authentication**

- [Authentication Strengths](#)
- [Enrollment Restrictions](#)

### **Browser SSO and Device Identity**

- [SSO Overview and Options for Apple Devices](#)
- [Configure Platform SSO for macOS](#)
- [Microsoft Enterprise SSO plug-in for Apple devices](#)
- [Configure macOS Enterprise SSO app extension \(Standalone\)](#)
- [Troubleshooting the Microsoft Enterprise SSO Extension](#)
- [Conditional Access Conditions \(Browser Support\)](#)
- [Platform SSO for macOS \(Apple Documentation\)](#)
- [Hybrid Azure AD joined devices show up as Unknown \(Q&A\)](#)