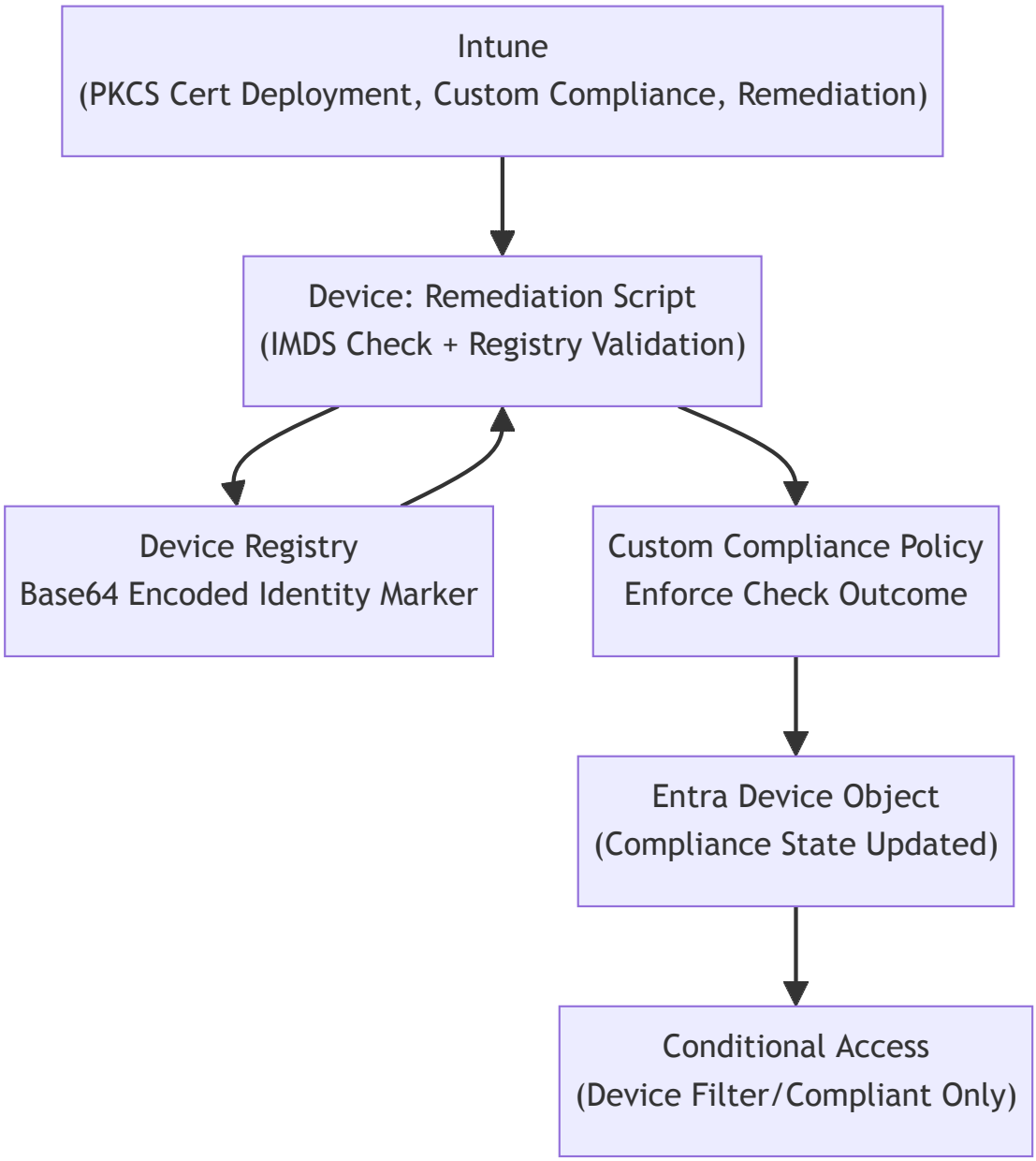
High-Level Design (HLD): SMDI Lite – Secure Device Metadata Ingestion using Intune Custom Compliance Policy

Objective

Mitigate the OuttaTune vulnerability without external automation layers (e.g. Logic Apps or Functions), by leveraging a continuously running Intune remediation script and custom compliance policy. This solution uses IMDS-based metadata checks, local registry verification, and Conditional Access to detect and enforce device legitimacy — particularly for Microsoft-hosted VMs (Dev Box / Windows 365).

Architecture Diagram



Flow Summary

1. **Device Enrollment via Intune**
   - PKCS certificate issued (optional).
   - Device receives Custom Compliance Policy and 15-minute interval remediation script.
2. **Remediation Script Logic**
   - Calls IMDS metadata endpoint.
   - If `azEnvironment = "AzurePublicCloud"`:
     - Parse `tags` or `tagsList` for Microsoft-hosted infra flag and customer tenant ID.
     - Check for expected Device Type (`Microsoft Dev Box` , `Cloud PC`).
   - Registry is checked/implanted with a **base64-encoded string** matching device type (based on `SystemProductName` and `SystemManufacturer`).
3. **Compliance Check**
   - Verifies expected device type against IMDS + system registry values.
   - If mismatch, mark device as **non-compliant**.
4. **Conditional Access**
   - Policies use device compliance state or require trusted user groups.
   - If a user's device fails compliance (e.g. DevBox or Cloud PC not validated), **access is blocked**.

Architecture Components

- **IMDS Metadata Endpoint**: Secure, internal source of cloud environment and tenant identity.
- **System Registry**: Stores expected device type in obfuscated (base64) format for integrity comparison.
- **Remediation Script**: Runs every 15 mins, validates cloud VM integrity and implants/compares registry marker.
- **Custom Compliance Policy**: Flags the device based on remediation script outcome.
- **Conditional Access Policies**: Allow/block access based on compliance + group membership.

Defence-in-Depth: Risk & Mitigation

Threat	Mitigation
On-device spoofing of device identity	IMDS data validated against known keys (e.g., `tags`, `azEnvironment`), and cross-checked with OS info.
Registry tampering	Registry path is obscure; value is base64-encoded. User tampering possible, but limited by script refresh every 15 minutes.
IMDS manipulation	IMDS is non-modifiable from within the guest OS.
Drift or environment change	Remediation script runs every 15 minutes, ensuring quick detection of drift or spoofing.
Use of unauthorized Dev Box / W365	Tenant mismatch in IMDS tags causes compliance failure.
Compliance policy misconfiguration	Script defines outcome; admin policy sets response (e.g., block via CA).
Overprivileged Dev Center user	Conditional Access blocks Dev Center users accessing cloud apps from non-compliant devices.
Logic App/Function App complexity/cost	Avoided entirely in this design — Intune native only.

Conditional Access Enforcement Scenarios

- Users in Dev Center group trying to access SharePoint from a non-compliant Dev Box → **Blocked**.
- Device moves from compliant to non-compliant after tampering → **Blocked within 15 minutes**.
- Attempt to spoof device model locally without IMDS tag → **Fails compliance check**.

Summary

Capability	Supported
Remote worker support	✅ Yes
Autopilot compatibility	✅ Yes
Tamper-resistance (basic)	✅ Yes
Custom Compliance Enforcement	✅ Yes
No external automation cost/complexity	✅ Yes
On-device write protection of CA inputs	⚠️ Partial – registry can be modified, but detected
PKCS optional (not central to this model)	✅ Optional
Continuous drift detection	✅ 15-min window
Cloud VM identification (DevBox/W365)	✅ Yes via IMDS