Scaling Intune Across Tenants with Microsoft Graph API

From manual portal work to automated, consistent multi-tenant management.

Sponsors

onedeploy
Windows Deployment Reimagined

ZERO TOUCH

Insight

control up

PATCH MY PC

robopack

numecent

Microsoft

eido

Rimo3

Maxime Guillemin

Maxime Guillemin
Microsoft MVP · Endpoint & Security

Role
Modern Workplace Architect

Focus
Intune · R&D · AI

Blog
https://cloudflow.be/

Joery Van den Bosch

Joery Van den Bosch
Microsoft MVP · Endpoint & Security

Role
Modern Workplace Architect

Focus
Intune · Security Copilot

Blog
https://intunestuff.com/

Agenda

Foundations

01 The Multi-Tenant Problem

02 Naming Conventions

03 Policy Layering

04 Microsoft Graph API

Operations

05 Backup

06 Templates as Code

07 Deploy & Drift

Delivery

08 Waved Rollout

09 Side-by-Side Updates

10 Takeaways & Q&A

Every MSP knows this feeling

You manage 5, 10, 20 tenants same baseline, but every one set up a little differently.

Inconsistent
Configurations drift across tenants. "We forgot to enable BitLocker on that one."

Slow onboarding
Days of repetitive portal clicking to stand up a new customer baseline.

Silent changes
A setting gets tweaked and nobody notices until a user calls in.

Knowledge silos
Config lives in people's heads not in code or documentation.

The portal doesn't scale. You need automation.

What we hoped vs. what we got

HOPED	REALITY
✓ Every tenant identical	✗ “We forgot BitLocker on that one”
✓ Onboard a new customer in hours	✗ Days of portal clicking, copy-paste errors
✓ Changes are tracked and documented	✗ Settings drift silently, nobody notices
✓ One dashboard, one report	✗ Log into 20 portals, compare by hand

vs.

The gap between the two columns is the work.

INTUNE TOOLKIT

If you can't read the name, you can't trust the config

The policy name is often the only thing linking a policy across tenants.

`{Version}` - `{Layer}` - `{PolicyType}` - `{PolicyName}`

`V2.0-MSP-BP-Compliance-BitLocker`

`V1.0-CIS-BP-EndpointSec-ASR`

`V1.0-CUST-BP-Config-CorporateWiFi`

MSP-BP
Your baseline. Every customer gets it.

CIS-BP
Opt-in hardening. Customer chooses.

CUST-BP
Customer-specific config.

Three layers, clear ownership

CUST-BP

Customer-Driven Config

Business-specific: Wi-Fi, VPN, app protection, industry compliance. Customer owns decisions, you advise.

CIS-BP

CIS Framework

CIS hardening layered on top. Opt-in per policy some controls break workflows. Customer chooses what to activate.

MSP-BP

MSP Baseline

Your functional baseline: BitLocker, Defender, Update rings. You build it, you maintain it. Every customer gets it.

overrides

Separation means clarity

01

“What did you change?”

Point to the exact layer and version. No ambiguity, no guessing.

Clarity

02

CIS benchmark update drops

Update CIS-BP across all tenants without touching MSP-BP or CUST-BP.

Isolated

03

Onboarding a new customer

Deploy Layer 1 + chosen Layer 2. Only Layer 3 is built from scratch.

Fast

04

Drift is detected

You instantly know if it's baseline, CIS, or customer-config drift each response differs.

Actionable

Your single interface to every tenant

GET `/configurationPolicies`
Settings Catalog

GET `/deviceCompliancePolicies`
Compliance

GET `/deviceConfigurations`
Config Profiles

POST `/deviceHealthScripts`
Proactive Remediations

POST `/mobileApps`
Applications

GET `/intents`
Endpoint Security

AUTH MODELS
Interactive · Device Code · Client Credentials

POWERSHELL
Microsoft.Graph.Authentication

ONE API
Every tenant, every policy type

INTUNE TOOLKIT

Connecting to any tenant, securely

Interactive
Browser popup

- Ad-hoc work
- Demos & troubleshooting
- Authenticate as yourself

Device Code
For headless environments

- SSH sessions
- CI/CD pipelines
- Enter code in any browser

Client Credentials
Service principal per tenant

- Fully automated
- Scheduled drift checks
- No user interaction

Security first: least privilege · certificate-based auth · one service principal per tenant · rotate on schedule

You can't compare what you haven't captured

ENDPOINT MANAGEMENT · 6 types

- ⚙ Settings Catalog
- ⚙ Device Configuration
- ✓ Compliance Policies
- ⚙ Endpoint Security
- ⚙ Group Policy (ADMX)
- > PowerShell Scripts

12 POLICY TYPES

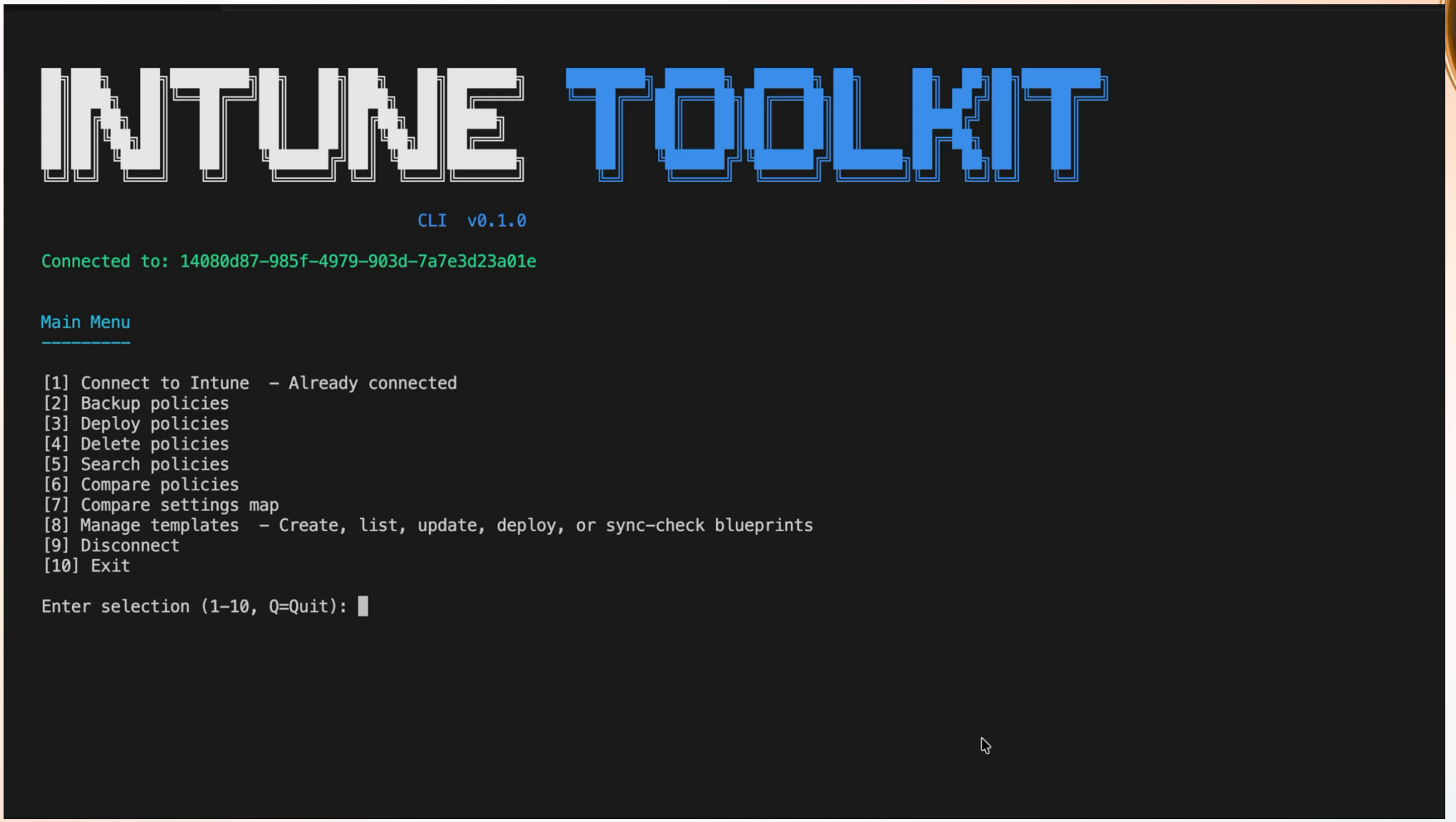
.json
BACK UP
tenant snapshot

APPS & PROVISIONING · 6 types

- ☐ Mobile Apps
- ⚙ App Config (Managed)
- ✓ App Protection
- ↻ Proactive Remediations
- → Autopilot Profiles
- > macOS Scripts

INTUNE TOOLKIT

Demo: Backup a tenant



Onboard a customer in minutes, not days

SOURCE

Template

DEPLOY

TARGET

Customer Tenant

policies created via Graph

Cross-tenant deploy uses `DisplayName` matching. Naming conventions pay off here.

Stop treating your baseline as a tenant

📁 template tree

```
templates/tmpl_<guid>/
└─ _metadata.json    # name · description · created
└─ v1.0/
   ├─ _version.json    # version notes · hash · count
   └─ policies/
      ├─ SettingsCatalog/
      ├─ DeviceCompliance/
      └─ EndpointSecurity/
└─ v2.0/    # next baseline version
```

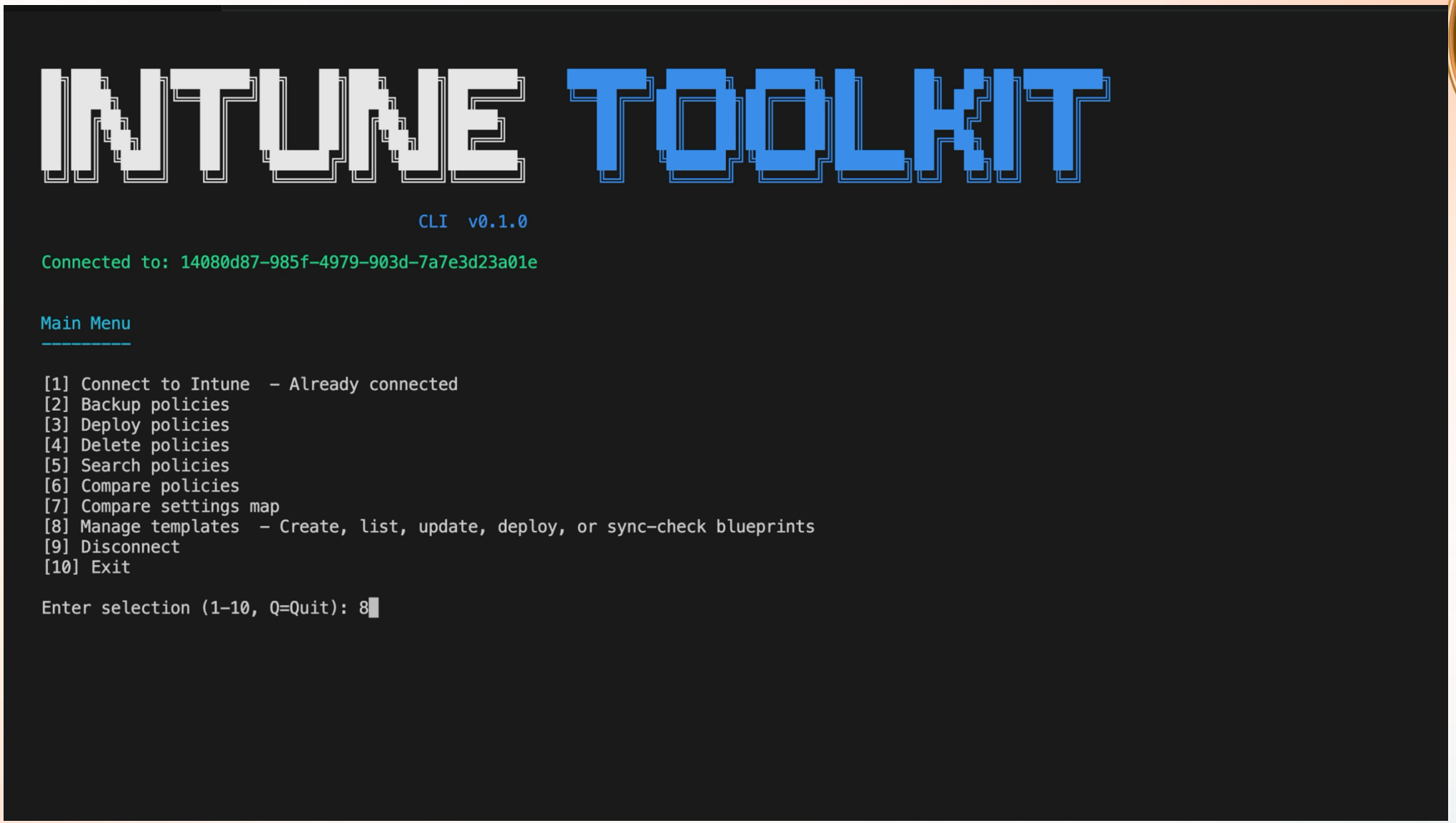
❌ **A "gold tenant"**
is a living thing. People change it by accident.

✅ **A template is a file**
versioned, immutable, portable, Git-friendly.

Versioned | **No assignments** | **Portable**

Source of truth:
a versioned file
not a tenant someone might change.

Demo: Templates



Templates are files put them in Git for full history and collaboration.

Demo: Deploy a baseline

Someone changed something. You need to know.

Configuration drift is the number-one silent killer in multi-tenant environments.

Drift within a tenant

Fresh backup vs. previous backup

- What changed since last week?
- Catches unauthorized or accidental changes.
- Your "is anything different from yesterday?" check.

Drift against a template

Tenant vs. MSP-BP or CIS-BP

- Is this customer still aligned with your baseline?
- Which policies have diverged from the template?
- Your compliance & governance check.

Name renamed?

Settings values modified?

Assignments groups changed?

Automate on a schedule. Catch drift before your phone rings.

Comparing against your baseline

Scenario: check is every tenant still aligned with your MSP baseline?

STEP 1 **Connect** to customer

STEP 2 **Backup** current state

STEP 3 **Compare** vs MSP-BP

STEP 4 **Review** drift report

STEP 5 **Remediate** or document

What the report shows

matched by DisplayName

Unchanged
Policy matches the template. All good.

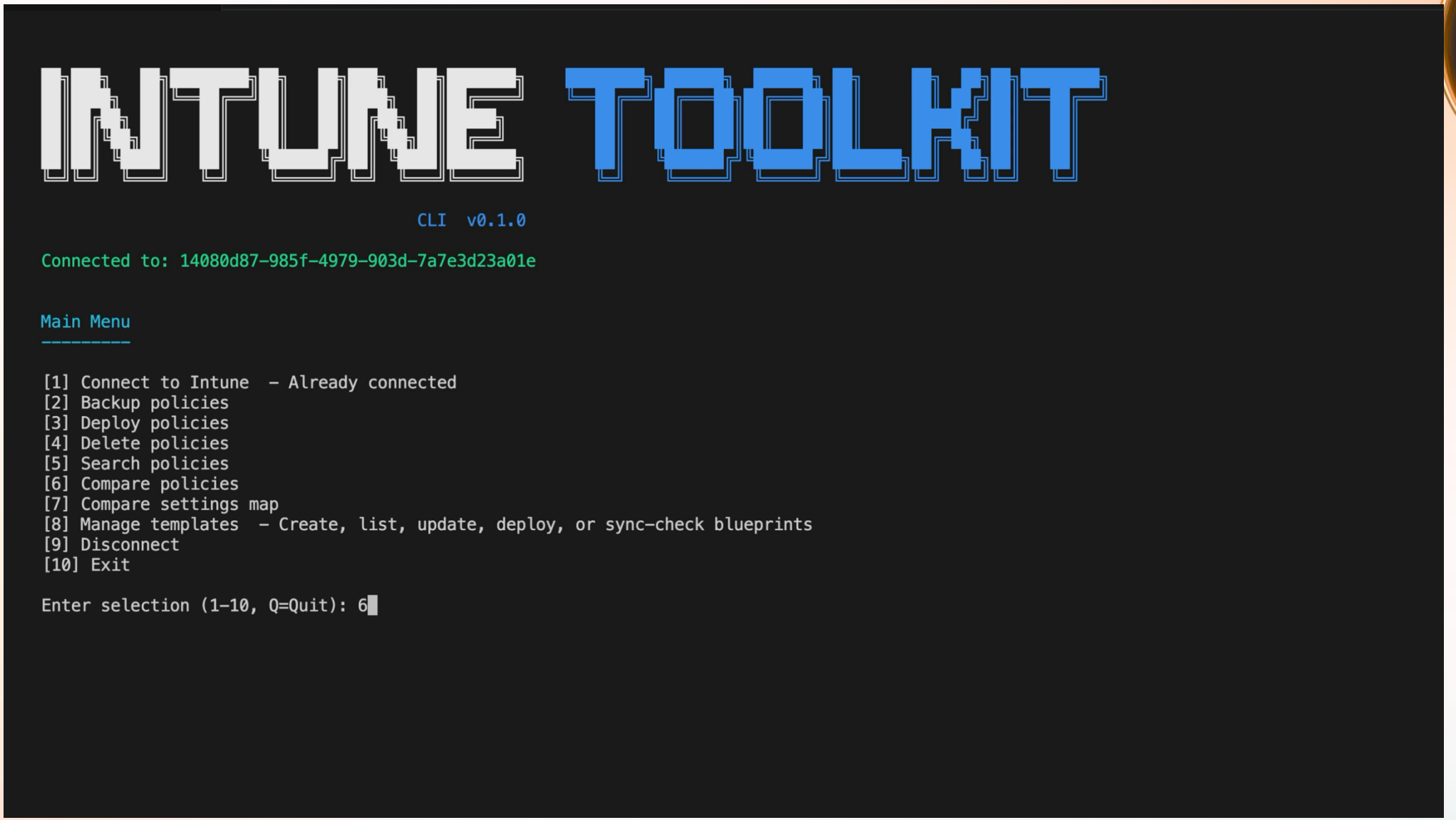
Changed
Settings differ. Investigate or remediate.

Missing in tenant
Template has it, tenant doesn't. Deploy it.

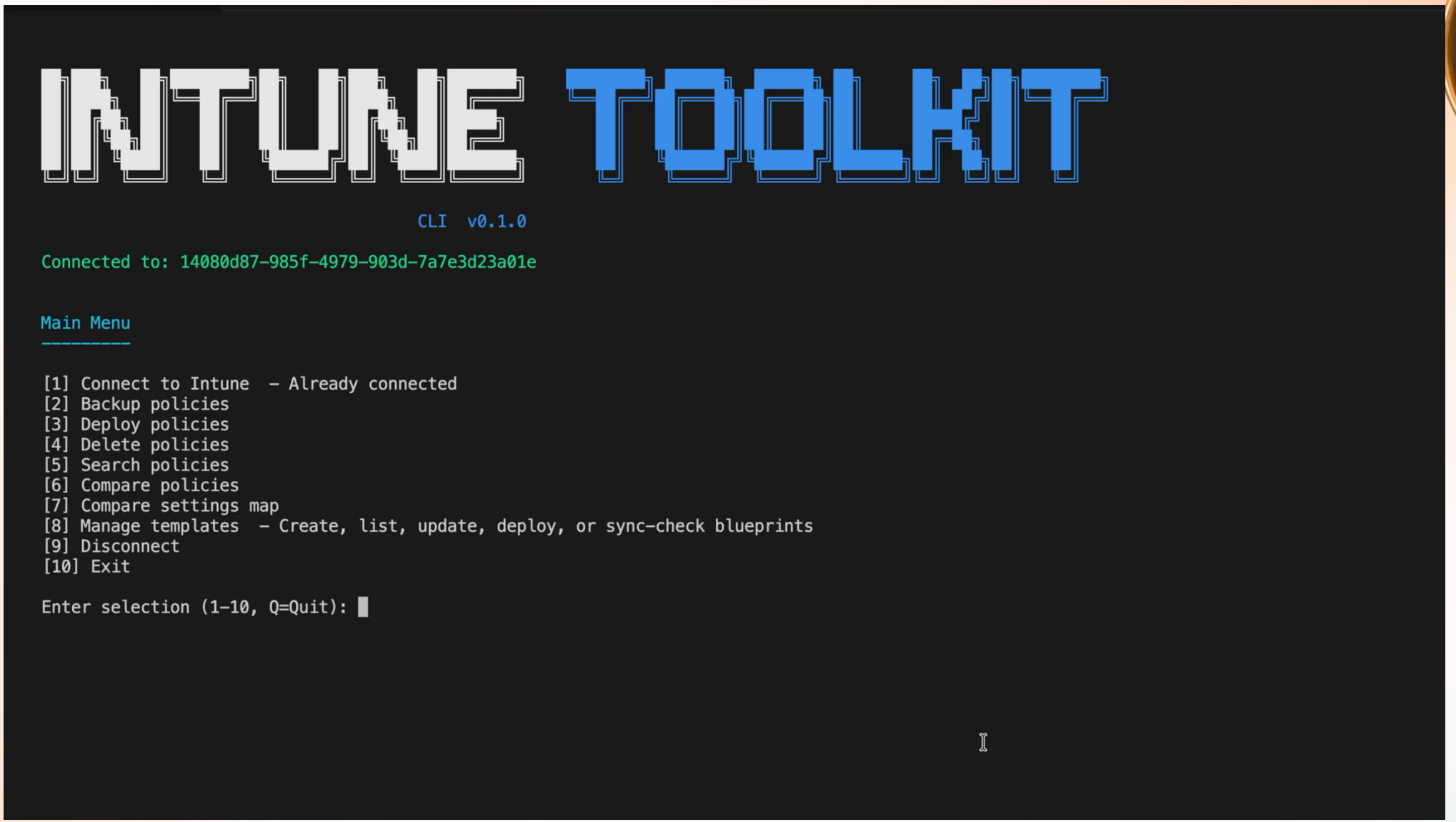
Extra in tenant
Is it CUST-BP or drift? Classify it.

DisplayName matching links policies across tenants. Naming convention = accurate drift detection.

Demo: Drift Check



Demo: Drift Check



Build once, version, deploy, compare

Scenario: you maintain a versioned MSP baseline, rolling changes out across every customer tenant.

1. Build
Baseline

2. Backup
Config to JSON

3. Tmpl
v1.0 created

4. Deploy
To customers

5. Evolve
Update → v2.0

6. Diff
Find delta

7. Push
Only changes

Why this cycle wins

Immutable source
Templates are files on disk. Nobody can change them by clicking in the portal.

Versioned history
v1.0 → v2.0 shows exactly what changed not just that something did.

Portable across tenants
Assignments stripped. Deploy anywhere without source-tenant artifacts.

Delta deploys only
Compare v1.0 vs v2.0 push only what changed. Minimal blast radius.

Your source of truth: a versioned file, not a tenant someone might change.

INTUNE TOOLKIT

Gradual assignment with rings

Staged Rollout Rings

20%
Ring 1

Pilot group

30%
Ring 2

Broad validation

50%
Ring 3

Full rollout

Ring Progression Rules

- ✓ Human approval between each ring no automatic advancement
- ✓ Issues at Ring 1 never reach 80% of users
- ✓ Each customer tenant progresses independently

Group naming: `SG_USR_{CustomerCode}_{Policy}_RING1/2/3`

Deploy new. Migrate assignments. Retire old.

v1.0 `V1.0-MSP-BP-Compliance-BitLocker`

Ring 1	Ring 2	Ring 3
assigned	assigned	assigned

All rings on the old version

Status: live, assigned, fully deployed.
Never edited in place.

migrate

v2.0 `V2.0-MSP-BP-Compliance-BitLocker`

Ring 1	Ring 2	Ring 3
assigned	*assigned*	*assigned*

Deployed, waiting for rings

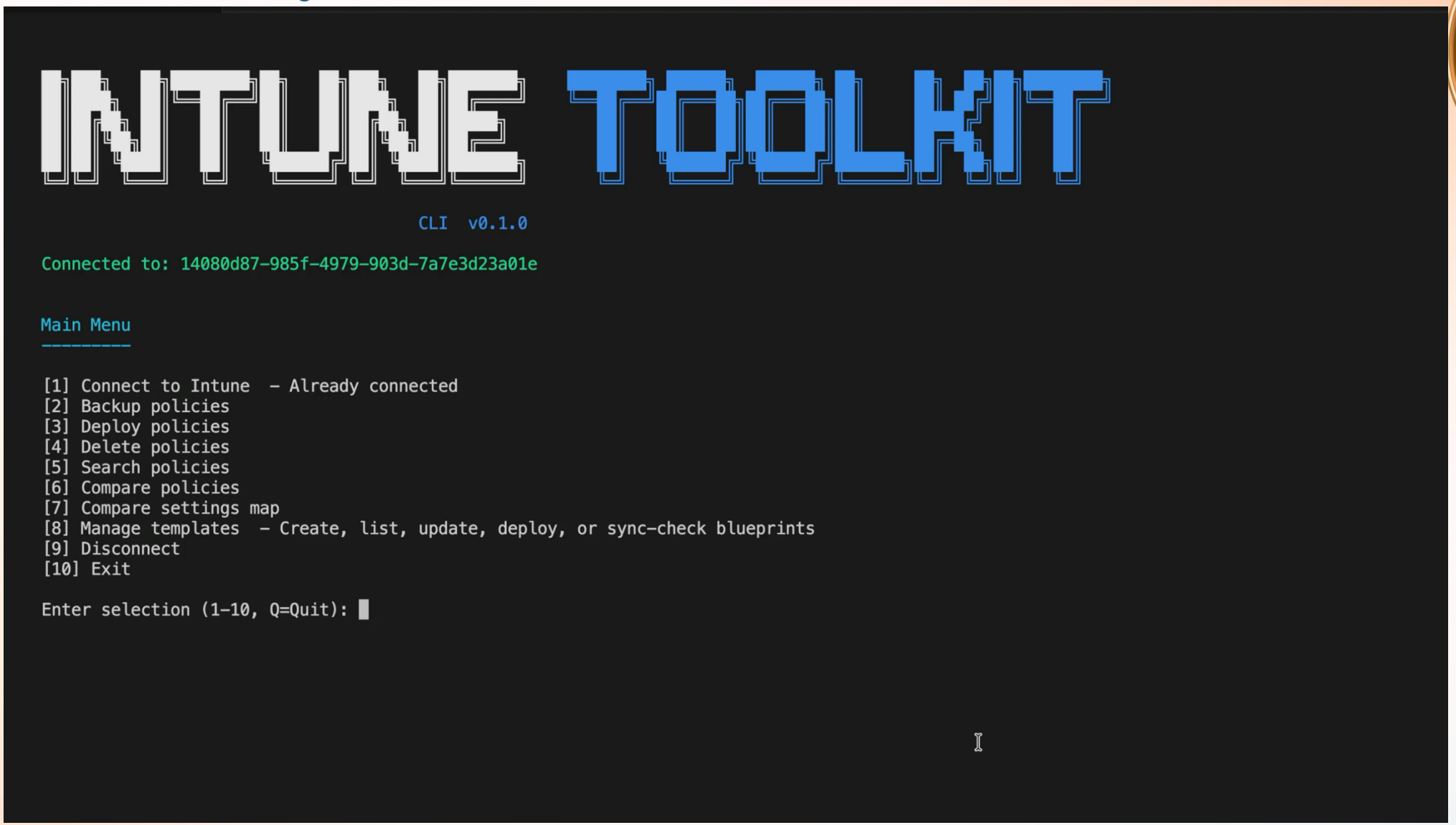
Status: deployed,

Never edit a live policy in place.

Deploy v2.0 → move Ring 1 → validate → promote Ring 2 → Ring 3 → retire v1.0.

Rollback = move an assignment, not reconstruct old settings.

Demo: Side-by-Side



A repeatable, scalable process

ONBOARDING
New customer

- Deploy MSP-BP template (latest).
- Deploy chosen CIS-BP policies.
- Build CUST-BP layer.
- Back up as initial baseline.

MAINTAINING
Existing customers

- Scheduled backups (weekly).
- Scheduled drift checks vs templates.
- Review drift remediate or document.
- Baseline update → new version → deploy delta.

INCIDENTS
Responding fast

- Compare current vs previous backup.
- Search: is this deployed everywhere?
- Redeploy from template to restore known-good.

One process. Every customer. Every week.

INTUNE TOOLKIT

Lessons from the field

01

Start with naming

Automation built on inconsistent names produces inconsistent results. Fix names first.

Foundation

02

Version your baselines

Don't “update” create v2.0. When something breaks, you need to know what changed.

Versioned

03

Treat policies as immutable

Never edit in place. New version, new policy. Migrate assignments. Retire the old one.

Immutable

04

Automate drift, not remediation

Start with scheduled compare-and-report. Trust the tool before you let it fix things.

Observe first

What you're taking home

1. **The portal doesn't scale.** Graph API is how you manage Intune across tenants.
2. **Naming is the foundation.** Cross-tenant compare relies on consistent policy names.
3. **Layer your policies.** MSP-BP · CIS-BP · CUST-BP clear ownership, clear versioning.
4. **Templates beat gold tenants.** A versioned file on disk > a tenant someone might change.
5. **Drift detection is the game-changer.** Catch problems before users report them.
6. **Never edit a live policy.** Side-by-side deploy, ring migration, retire the old.
7. **Automation is a journey.** Backup & compare → deploy → full auto.

Questions?

Thank you