Young Consulting Group | Secure Stack Monthly
www.youngconsultinggrouop.co
admin@youngconsultinggroup.co

Secure Stack Monthly

Issue #3 – Configuration Management Foundations

Intro

Configuration Management (CM) is where your system's "secure by design" claims either hold up — or fall apart. Auditors want to see proof that you know what's installed, how changes are tracked, and that baselines are maintained. In this issue, we'll cover **CM-2 (Baseline Configuration), CM-3 (Configuration Change Control), and CM-6 (Configuration Settings).** Together, these controls create the backbone of system hardening and operational discipline.

CM-2: Baseline Configuration

What it means:
Every system must have a documented baseline — a reference state that defines secure and approved configurations.

Implementation checklist:

- Build from hardened images (STIG/CIS benchmarks).
- Document baseline OS and application versions.
- Store baselines in a version-controlled repo (Git, Ansible, SCCM).
- Update baseline whenever patches or upgrades are applied.
- Retain baseline history for audit reference.

Audit artifacts:

- Baseline documentation (Word, PDF, or code repo).
- Image build scripts (Kickstart, Ansible, PowerShell DSC).
- Evidence of baseline reviews.

Common pitfall:
Teams apply hardening but never document it — leaving "tribal knowledge" instead of artifacts.

CM-3: Configuration Change Control

What it means:
All changes to system configurations must follow an approval and review process. No silent changes.

Implementation checklist:

- Enforce Change Control Board (CCB) or ticket approval for changes.
- Document risk impact of each change.
- Test changes in staging before production.
- Maintain rollback procedures.
- Update SSP/diagrams when changes affect security posture.

Audit artifacts:

- Change tickets (ServiceNow, Jira, Remedy).
- CCB minutes/approvals.
- Evidence of rollback tests.

Common pitfall:
Admins applying "quick fixes" in production without approval — auditors will classify this as a CM-3 violation.

CM-6: Configuration Settings

What it means:
Systems must implement mandatory security settings aligned with baselines and enforce them consistently.

Implementation checklist:

- Apply STIG or CIS hardening guides.
- Automate enforcement with Ansible, Puppet, or Group Policy.
- Periodically re-verify system configs against baselines.
- Document any deviations with Risk-Based Decisions (RBDs).

Audit artifacts:

- STIG Viewer output or CIS benchmark reports.
- Ansible/Puppet/GPO config scripts.
- Waiver documentation for deviations.

Common pitfall:
Hardening applied once, then forgotten — systems drift back to insecure defaults.

Case Study: The Drifted Baseline

A federal contractor's Linux servers passed initial ATO inspection but failed during a surprise review six months later. A configuration scan revealed several STIG settings were no longer enforced.

- **Problem**: CM-2 baseline not updated, CM-6 settings not enforced.
- **Impact**: Immediate POA&M issued, delay in renewal.
- **Fix**: Implemented automated Ansible playbooks to enforce STIG settings nightly, created quarterly baseline review SOP.

Lesson: Without automation, secure configurations will drift — and auditors *will* catch it.

Closing & Next Issue

Key takeaways:

1. Baselines must be documented and version-controlled (CM-2).
2. All changes must be reviewed and approved (CM-3).
3. Hardening settings must be enforced continuously, not once (CM-6).

Next Issue Preview: We'll cover **Identification & Authentication (IA-2, IA-5, IA-8)** — how to lock down identity, passwords, and MFA across federal environments.

Check out our publications available on Amazon and our website:

Secure the Shell
Fixing Real-World Linux
THE ISSO PLAYBOOK
Acing the Linux Admin Interview
A Linux Engineer's Guide to Speaking NIST 800-53
Amazon
Website