

Lab Guide

Four exercises: SSH config audit, Linux baseline, Windows GPO review, and Dockerfile hardening — turn AI into a configuration reviewer.

Authorized Use Only

All configs are **synthetic**. Apply the hardening patterns only to systems you own and have tested in staging first.

Prerequisites

- Completed the [Theory](#) reading
- Access to an AI assistant (Claude, ChatGPT, or local LLM)
- Downloaded all four files into a folder named `harden-lab/`
- Familiarity with: sshd_config, sysctl, Windows GPO basics, Docker

Lab Scenario

AcmeCorp is preparing for a compliance audit (CIS + PCI). You have pulled configuration snapshots from a sampling of hosts. Your job: identify gaps against CIS Benchmarks, write the corrected settings, and produce a prioritized remediation plan that does not break production.

Download Lab Files

Save all 4 files into a folder named `harden-lab/` before starting the exercises.

↓ [dockerfile-insecure.txt](#) ~1 KB

↓ [linux-audit.txt](#) ~9 KB

↓ [sshd-config-weak.txt](#) ~2 KB

↓ [windows-gpo-audit.txt](#) ~12 KB

Exercise 1 — SSH Config Audit

Open [sshd-config-weak.txt](#). SSH is the front door to every Linux host.

Task 1.1 — Score it against CIS

Below is `/etc/ssh/sshd_config`. Audit against CIS Benchmark:

1. `PermitRootLogin` (should be no)
2. `PasswordAuthentication` (should be no if keys are available)
3. `PermitEmptyPasswords`, `ChallengeResponseAuthentication`
4. Protocol version, `KexAlgorithms`, `Ciphers`, `MACs`
5. `MaxAuthTries`, `LoginGraceTime`, `ClientAlive*`
6. `AllowUsers` / `AllowGroups` allowlisting
7. `X11Forwarding`, `AllowAgentForwarding`, `AllowTcpForwarding`
8. `LogLevel` (should be INFO or VERBOSE)
9. `UsePAM`, `Subsystem sftp`

For each finding:

- CIS control number
- Severity
- Current line
- Corrected line
- Rollout risk (will this break existing users?)

Output the full corrected `sshd_config` at the end.

[paste `sshd-config-weak.txt`]

✓ **Self-check: Your corrected config should be paste-ready and include at least one `AllowGroups`-style allowlist.**

Exercise 2 — Linux System Audit

Open `linux-audit.txt`. This is the output of a host inventory script.

Task 2.1 — Flag the exposures

Below is a Linux system audit snapshot. Review for:

1. Kernel / package versions with known CVEs.
2. Users with UID 0 other than root.
3. SUID/SGID binaries outside the default set.
4. World-writable files in /etc, /usr, /var.
5. Listening services on 0.0.0.0 that should be localhost.
6. sysctl hardening gaps (net.ipv4.conf.*.rp_filter, kptr_restrict, dmesg_restrict, unprivileged_usersns_clone).
7. Audit/auditd not configured, or rules missing.
8. Sudoers NOPASSWD abuse, ALL=(ALL) blanket rules.
9. Firewall (ufw/firewalld/nftables) disabled.

Output a prioritized remediation list: [critical / high / medium] with the fix command for each.

[paste linux-audit.txt]

✓ Self-check: Your list should have at least 3 items that map directly to a CIS control number.

Exercise 3 — Windows GPO Review

Open `windows-gpo-audit.txt`. Group Policy is the baseline for a Windows fleet.

Task 3.1 — Harden the policy

Below is a Windows GPO export / RSOP dump. Audit:

1. Password policy (length, complexity, age, history, lockout).
2. Kerberos policy (ticket lifetime, renewal).
3. Audit policy (logon/logoff, object access, privilege use).

4. User rights assignment (Debug programs, Backup files, Impersonate a client).
5. Security options (LM hash storage, SMBv1 enabled, NTLM level).
6. Windows Defender / AV configuration.
7. BitLocker, Credential Guard, Device Guard state.
8. PowerShell execution policy and script block logging.
9. LAPS deployment (local admin password solution).

For each gap: the CIS control, the registry/GPO path, the value to set, and the impact on users.

[paste windows-gpo-audit.txt]

✓ Self-check: You should find at least one legacy protocol (SMBv1, NTLMv1, LM hash) that must be disabled.

Exercise 4 — Dockerfile Hardening

Open `dockerfile-insecure.txt`. Containers are only as safe as their base image and runtime flags.

Task 4.1 — Rewrite it securely

Below is a Dockerfile. Review for:

1. Base image: is it pinned by digest? Is it minimal (distroless, alpine)? Is it current?
2. USER directive (no root).
3. Unnecessary packages installed.
4. Secrets baked into layers.
5. COPY vs ADD misuse; overly broad COPY . .
6. Ports exposed that should not be.
7. HEALTHCHECK presence.
8. Build-time vs run-time separation (multi-stage builds).
9. Entrypoint / CMD abuse (shell form vs exec form).

Output:

- Findings list with severity
- The rewritten Dockerfile (paste-ready)
- A matching docker run / compose snippet with `--read-only`,

```
--cap-drop=ALL, --security-opt=no-new-privileges
```

```
[paste dockerfile-insecure.txt]
```

✓ Self-check: Your rewritten Dockerfile should be multi-stage, non-root, and at least 40% smaller.

Final Exercise — Rollout Plan

Task 5.1 — One-week hardening sprint

Combine the four audits into a one-week hardening rollout plan:

- Day 1: Low-risk, high-impact (SSH, sysctl, audit rules)
- Day 2-3: Medium risk (GPO rollout to pilot OU)
- Day 4: Container rebuilds
- Day 5: Full fleet rollout + monitoring

For each day list: what changes, how to verify, rollback plan, communication to users.

Format as markdown, max 1500 words.

✓ Self-check: Each day should have a verification step and a rollback.

What's Next

Lesson 05 moves from host to **network policy** — ACLs, iptables, and cloud security groups.