

# Lab Guide

Four exercises: network diagram review, ACL audit, iptables analysis, and AWS security group review — enforce least-privilege at the network layer.

## Authorized Use Only

All policies are **synthetic**. Do not apply generated rules directly to production without a change window and rollback.

## Prerequisites

- Completed the [Theory](#) reading
- Access to an AI assistant (Claude, ChatGPT, or local LLM)
- Downloaded all four files into a folder named `firewall-lab/`
- Familiarity with: TCP/IP, stateful firewalls, iptables syntax, AWS SG model

## Lab Scenario

AcmeCorp's network has grown organically. You have the logical diagram, a Cisco-style ACL, an iptables dump from a Linux bastion, and a set of AWS security groups. Your mission: find over-permissive rules, collapse duplication, and enforce least-privilege without breaking the apps.

### Download Lab Files

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

↓ [acl-audit.txt](#) ~7 KB

↓ [aws-security-groups.txt](#) ~6 KB

↓ [iptables-rules.txt](#) ~3 KB

↓ [network-diagram.txt](#) ~7 KB

## Exercise 1 — Network Diagram Review

Open [network-diagram.txt](#). A text-based diagram is still a diagram — read it as topology.

### Task 1.1 — Map trust zones

Below is a logical network diagram. Produce:

1. The trust zones (internet, DMZ, app, db, admin, OT).
2. For each zone: what should be ALLOWED to enter, what should leave, and what should NEVER cross the boundary.
3. Single points of failure or flat segments that collapse zones.
4. Any asset in the wrong zone (e.g., admin jump host in DMZ).
5. Management plane paths — are they isolated?
6. Backup / DR paths — do they create a bypass?

Output: a zone matrix (source -> destination) with allow/deny.

[paste network-diagram.txt]

✓ **Self-check:** You should be able to state in one sentence the principle of segmentation this network violates most.

## Exercise 2 — ACL Audit

Open [acl-audit.txt](#). Cisco-style ACLs get stale fast.

## Task 2.1 — Collapse and tighten

Below is a Cisco extended ACL. Audit for:

1. "permit ip any any" or similar catch-alls.
2. Shadowed rules (a later rule is unreachable).
3. Duplicate rules covering the same traffic.
4. Rules missing a log directive on deny.
5. Ports opened to 0.0.0.0/0 that should be narrow.
6. Management protocols (SSH, SNMP, Telnet) open to non-admin nets.
7. ICMP types: allowed everywhere?
8. Ordering: most-specific first, least-specific last.

Output the refactored ACL:

- Grouped by purpose
- Commented with intent per rule
- With explicit deny + log at the end

[paste acl-audit.txt]

✓ **Self-check: Your refactored ACL should be shorter and more readable than the original.**

## Exercise 3 — iptables Review

Open `iptables-rules.txt`. iptables is powerful but easy to misconfigure.

### Task 3.1 — Rewrite in least privilege

Below is an iptables-save dump. Review for:

1. Default policies (INPUT, OUTPUT, FORWARD) — should be DROP.
2. Stateful rule (ESTABLISHED,RELATED) present early?
3. Loopback allowed (important!).
4. Any ACCEPT rule for 0.0.0.0/0 on a sensitive port.
5. NAT rules that expose internal services.
6. Unneeded custom chains.
7. Missing rate-limiting on SSH / exposed services.

8. IPv6 parity (iptables equivalent rules?).

Output:

- Findings list
- Rewritten ruleset (iptables-restore format)
- The two-line command to apply it with rollback timer

[paste iptables-rules.txt]

✓ **Self-check: Your ruleset should default-drop and allow only explicit services.**

## Exercise 4 — AWS Security Groups

Open `aws-security-groups.txt`. Cloud SGs are the most common source of "open S3 / open DB" incidents.

### Task 4.1 — Tighten the cloud perimeter

Below is a list of AWS security groups with rules. Audit for:

1. Any rule with 0.0.0.0/0 on a non-public port (22, 3306, 5432, 3389, 27017, 9200).
2. SGs that reference other SGs overly broadly.
3. Rules with port range 0-65535 ("any port").
4. IPv6 :::/0 parity issues.
5. Unused SGs (hint: count ENIs).
6. SGs with descriptive names that lie ("temp-debug-2019").
7. Default SG usage (should be empty and unused).

For each finding:

- SG ID / rule index
- CVSS-like impact
- Exact terraform / CLI to fix
- What the app might break and how to verify

[paste aws-security-groups.txt]

✓ Self-check: You should find at least one rule that exposes a database or admin port to the internet.

## Final Exercise — Unified Policy Doc

### Task 5.1 — Cross-layer firewall strategy

Using the four audits, produce a unified firewall policy for AcmeCorp:

1. Zone model (from exercise 1)
2. Default-deny everywhere
3. Per-zone allow matrix
4. Change-control process for new rules
5. A quarterly audit checklist

Format as markdown, max 1500 words.

✓ Self-check: The policy should be short enough to be printed on one page as a reference for network engineers.

### What's Next

Lesson 06 dives into **secrets and crypto hygiene** — TLS configs, leaked credentials, and home-grown crypto gone wrong.