DOWNLOADABLE ENGINEERING REFERENCE / PDF EDITION

API Authorization Review Notes

Authorization failures are rarely “missing auth.” They are usually boundary failures: inconsistent enforcement, privilege drift, and implicit trust in client-provided inputs.

These notes outline a practical authorization review that teams can run before production exposure becomes incident response.

1) Identify Enforcement Points

- List where authorization is enforced today:
  - API gateway
  - service layer
  - database layer
  - worker/queue consumers
- Confirm enforcement is consistent across:
  - REST/GraphQL APIs
  - internal admin tools
  - background jobs
  - webhooks and inbound integrations

2) Define Invariants Per Resource

For each high-value resource, define invariants explicitly:

- who can read
- who can write
- who can delete
- what “ownership” means (and where it is enforced)
- which admin operations exist and how they are guarded

3) Boundary Checks That Frequently Fail

- object-level authorization vs “account-level only”
- multi-tenant scoping under edge cases
- privilege escalation paths via:
  - IDOR-style access patterns
  - weak “role” checks
  - admin workflows that bypass policy
- “soft delete” and historical data access behavior

4) Abuse Resistance (Not Just Auth)

Authorization may be correct and still exploitable under abuse:

- endpoint enumeration
- token farming and replay
- rate limit bypass
- automation against account recovery flows
- resource exhaustion via unbounded queries or retries

5) Auditability and Incident Reconstruction

If an incident happens, you should be able to answer:

- what resource was accessed
- by whom (principal identity)
- through which path (API, worker, integration)
- why it was allowed (policy)

Avoid logging raw secrets. Prefer:

- correlation IDs
- policy identifiers / reason codes
- stable principal identifiers

6) Outputs

Practical outputs that engineers can act on:

- a trust-boundary map with enforcement points
- a prioritized risk list (impact and exploitability)
- a remediation plan tied to systems constraints
- a validation plan (tests + monitoring signals)

Public engineering reference, not a completed assessment, legal advice, audit, certification, or customer deliverable.

Editable Markdown source: demonicbinary.com/downloads/api-authorization-review-notes.md