API security: is every request authorized for the exact object it touches?

Check in code, on every request, that the caller may act on the exact object and fields it names, and add tests this month that replay each endpoint with a second tenant's token. Use the gateway for token checks, rate limits, schema checks and the version inventory, and switch off any API that isn't in that inventory.

Reviewed **4 Nov 2026** Next review **4 Dec 2026** Version **1.0**

Written for

Developers, architects and platform teams that build web, mobile or partner APIs (REST, GraphQL, gRPC, webhooks), and the security staff who review them

Latest version

sheets.einblick.dev/api-security/

If you only do five things

1. **Authorize every object in the handler.** Look up records by tenant and ID together. OWASP calls broken object level authorization extremely common[1]. In July 2026 CISA listed as exploited a Langflow flaw that let users run other users' flows by ID[14].
2. **Accept and return only allowlisted fields.** Bind each request to an input schema per endpoint, and build responses from an explicit output type, not a generic `to_json()` of the database object[2].
3. **Validate every token fully at the API.** Check signature, algorithm, issuer, audience, expiry and scope[5]. In 2026 attackers exploited SimpleHelp skipping OIDC signature checks and JFrog Artifactory ignoring token scope[14].
4. **Inventory every API and switch off the rest.** List each host, version and owner, including beta and partner hosts, and block retired versions. The NCSC traces the Optus breach to a forgotten API that stayed online for years[10].
5. **Test with a second tenant before each release.** Replay every call with another tenant's token, a lower role and no token, and block the release on any success[1].

Test an API for authorization flows

1. **Set up two tenants.** Give each an admin, a basic user and one object of every type. CISA advises testing with users of different privilege levels[13].
2. **Collect every call.** Take them from the spec, the gateway and recorded web and mobile traffic, including old versions and GraphQL operations.
3. **Swap the caller.** Send each request with the other tenant's token, the basic user's token and no token. Expect `401`, `403` or `404`.
4. **Swap the IDs and fields.** Change IDs in paths, bodies and batches. Add `role`, `tenantId` or `isAdmin` to writes and check they are ignored.
5. **Keep each case as a CI test.** Run them for every route. NIST suggests deploying an endpoint only once it carries a permission annotation[9].

How API attacks work now

- **Logged-in users reach other users' data.** In OWASP's 2025 Top 10 data, 100% of applications tested had some form of broken access control, a category that also covers SSRF and CSRF[4]. Three of the API Top 10's first five risks are about authorization[9].
- **Old and side-door endpoints let attackers in.** OWASP says unmanaged APIs are usually breached via old versions or unpatched endpoints[3]. The NCSC says weak authentication and no rate limiting in Dell's partner portal API exposed details of 49 million customers[10].

Tokens and sign-in

- **Use the code flow with PKCE.** RFC 9700 forbids the password grant, advises against the implicit grant and requires PKCE for public clients[6].
- **Bind each token to one API.** Restrict access tokens to one audience and the least privilege; an API must refuse tokens meant for another[6]. Identify users by `iss` plus `sub`[5].
- **Sender-constrain tokens, high-value ones first.** RFC 9700 recommends mutual TLS or DPoP for all access tokens, and requires sender-constrained or rotated refresh tokens for public clients[6].
- **Keep lifetimes short.** Authorization codes last at most 10 minutes and refresh tokens need an absolute expiry[5]. Keep access tokens to minutes. Give outside callers opaque tokens, not JWTs that reveal internal claims[9].
- **Don't rely on API keys alone.** The NCSC calls them shared bearer tokens, often broad and without expiry[11]. Authenticate partner and service clients with `private_key_jwt` or mutual TLS[6].

Gateway or code: what goes where

- **The gateway checks who and how much.** Validate tokens, reject requests outside the schema, cap payload size and time, and log every call[9,12].
- **The code checks which object.** Ownership, tenant, record state and field rules need the data, so the service decides, or a policy service it asks.
- **Make the gateway impossible to skip.** Accept traffic only from the gateway, by network policy or a credential it attaches[9]. In January 2026 CISA listed a Versa Concerto proxy flaw that exposed admin endpoints[14].
- **Strip identity headers at the edge.** Clients must not be able to set headers such as `X-User-ID` that the gateway sets[5].
- **Rate-limit per caller.** Key limits to the user or credential over 1 to 5 minutes and cap concurrent requests; a total limit lets one caller crowd out the rest[9].

Webhooks, GraphQL and gRPC

- **Verify every webhook you receive.** Check the HMAC over the raw body in constant time[17,18], reject old timestamps (Stripe's libraries allow 5 minutes by default) and skip event IDs already processed[18].
- **Treat webhook URLs you call as hostile.** CISA listed a GitLab webhook SSRF as exploited in February 2026[14]. Allowlist hosts, check the address you connect to and refuse redirects, as the TypeScript sheet explains.
- **Cap GraphQL query cost.** Use depth and amount limits or cost analysis[5], limit batching and check authorization on both edges and nodes[15].
- **Hide the schema on public endpoints.** Turn off introspection unless outside parties use the API[5], and GraphiQL in production[15]. gRPC's docs flag the reflection service as a possible security issue on public APIs[16].

What changed in 2026

- **Browser apps should keep tokens on a backend.** RFC 10017, a Best Current Practice since August 2026, strongly recommends a backend for frontend, which keeps tokens out of the browser, for business apps and apps that handle personal data[7].
- **Agent-facing APIs need the same checks.** MCP's July 2026 specification deprecates Dynamic Client Registration[19], and in September CISA listed a LiteLLM MCP endpoint that accepted any bearer token[14]. Inventory and test MCP servers like any other API.
- **NIST's API guidance gained checklists.** The March 2026 update of SP 800-228 adds tables of API risks and of controls by lifecycle stage[9].

Mistakes that cost the most

- **Checking the role but not the record.** NIST counts a missing resource-level check, or one against the wrong user or resource, as an authorization risk[9].
- **Letting the app hide fields.** Attackers read the raw API, and decompile mobile apps to find endpoints and keys[12].
- **Leaving a beta host behind.** In OWASP's example, a rate limit in front of the production reset API didn't cover a beta host running the same code, where a researcher brute-forced the 6-digit token[3].
- **Calling other services with the backend's own token.** Pass the caller on by token exchange and base each decision on their permissions[5], or one bug in the first service reaches every service behind it.

What you can skip

- **Waiting for OAuth 2.1.** It is still a draft, at version 17 in October 2026[8]. RFC 9700 already sets the rules[6].
- **Random IDs instead of checks.** CISA says UUIDs and GUIDs should not be used for security capabilities[13]. Use them, and check access anyway.
- **An API security platform before an inventory and tests.** No traffic product knows that order 1042 belongs to tenant B.
- **Signing every message.** ASVS asks for per-message signatures only at its highest level, for highly sensitive requests[5].

What this looks like at your size

Small team, no security staff

- Write one `canAccess(user, object, action)` check and call it in every handler that takes an ID.
- Use a hosted identity provider with the code flow and PKCE.
- Put the API behind your cloud's managed gateway with per-key rate limits.

Growing, with IT or security staff

- Generate the inventory from specs and gateway routes, with owners and retirement dates.
- Run the two-tenant tests in CI, and turn off introspection and reflection by default.
- Alert on bursts of 403 and 404 from one caller, a sign of enumeration[13].

Enterprise or regulated

- Annotate each endpoint with its permission and enforce it at the gateway via a policy service[9].
- Require sender-constrained tokens for payment, health and admin APIs[5].
- Pen-test each major release with users of different roles in different tenants.

Version history

4 Nov 2026, version 1.0 Published after a copy edit and a source-by-source fact check, which narrowed 7 statements to what the sources say.

Sources

1. API1:2023 Broken Object Level Authorization. OWASP, 2023. api-security.owasp.org
2. API3:2023 Broken Object Property Level Authorization. OWASP, 2023. api-security.owasp.org
3. API9:2023 Improper Inventory Management. OWASP, 2023. api-security.owasp.org
4. A01:2025 Broken Access Control. OWASP Top 10, 2025. top10.owasp.org
5. ASVS 5.0.0, chapters V4 and V8 to V10. OWASP, 30 May 2025. github.com
6. RFC 9700, Best Current Practice for OAuth 2.0 Security. IETF, Jan 2025. rfc-editor.org
7. RFC 10017, OAuth 2.0 for Browser-Based Applications. IETF, Aug 2026. rfc-editor.org
8. The OAuth 2.1 Authorization Framework, draft 17. IETF, 31 Oct 2026. datatracker.ietf.org
9. SP 800-228 upd1, Guidelines for API Protection for Cloud-Native Systems. NIST, 13 Mar 2026. csrc.nist.gov
10. New guidance on securing HTTP-based APIs. NCSC (UK), 3 Apr 2025. ncsc.gov.uk
11. Securing HTTP-based APIs: authentication and authorisation. NCSC (UK), 3 Apr 2025. ncsc.gov.uk
12. Securing HTTP-based APIs: limiting exposure. NCSC (UK), 3 Apr 2025. ncsc.gov.uk
13. Preventing Web Application Access Control Abuse (AA23-208A). ACSC, CISA and NSA, 27 Jul 2023. cisa.gov
14. Known Exploited Vulnerabilities Catalog. CISA, 3 Nov 2026. cisa.gov
15. GraphQL Cheat Sheet. OWASP, checked 4 Nov 2026. cheatsheetseries.owasp.org
16. Reflection. gRPC docs, 11 May 2026. grpc.io
17. Validating webhook deliveries. GitHub Docs, checked 4 Nov 2026. docs.github.com
18. Receive Stripe events in your webhook endpoint. Stripe Docs, checked 4 Nov 2026. docs.stripe.com
19. Key Changes, specification 2026-07-28. Model Context Protocol, 28 Jul 2026. modelcontextprotocol.io