CS-071 · API Security

API Gateways

A single controlled entry point that authenticates, routes, and protects traffic to back-end services.

Executive Summary

An API gateway is a single, controlled entry point that sits in front of back-end services and handles cross-cutting concerns such as authentication, authorization, rate limiting, routing, and logging. It lets teams enforce consistent security in one place instead of rebuilding it in every service. A gateway is powerful but also a high-value target, so it must be hardened and must never be treated as the only line of defense.

What It Is

An API gateway is a reverse proxy purpose-built for APIs. Clients send their requests to the gateway rather than directly to individual services, and the gateway decides what to do with each request: verify the caller's token, enforce rate limits and quotas, route the request to the right back-end service, transform or aggregate responses, and record logs and metrics. This centralizes concerns that would otherwise be duplicated across many services, which is especially valuable in a microservices architecture where dozens of services would each need the same security logic. The gateway becomes the enforcement point for policy at the edge, giving a consistent place to apply authentication, authorization checks, and traffic controls before requests ever reach internal systems.

Why It Matters

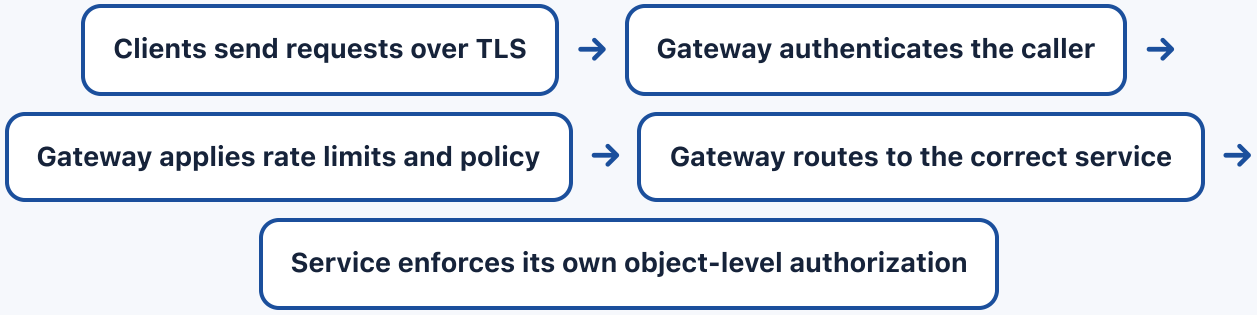
As organizations split systems into many services and expose them to mobile apps, partners, and single-page front ends, consistent edge security becomes critical. An API gateway makes it practical to apply the same authentication, rate limiting, and logging across every endpoint, which reduces the chance that one forgotten service ships without protection. It also gives

security teams a single place to observe traffic and respond to abuse. At the same time, the gateway concentrates risk: because all traffic flows through it, a misconfiguration or compromise can expose every service behind it. For engineers and security professionals, understanding gateways is essential both to enforce policy efficiently and to avoid creating a single point of failure.

How It Works

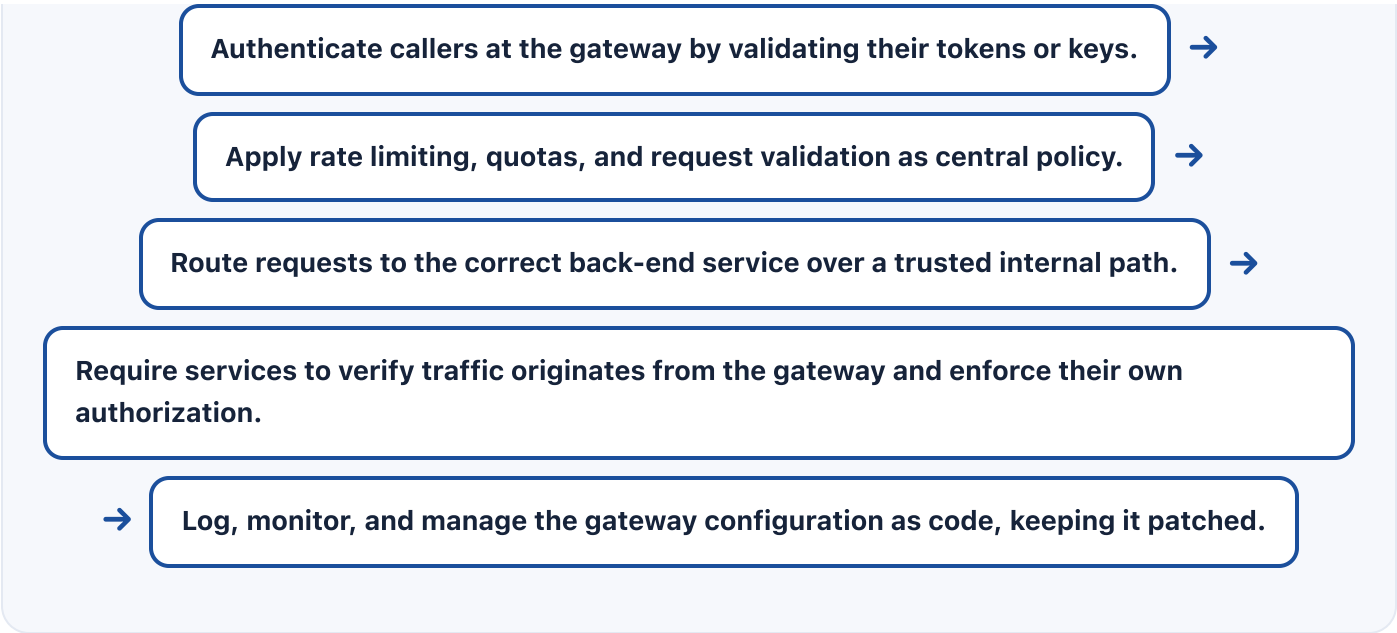
A request first reaches the gateway, usually over TLS. The gateway authenticates the caller, commonly by validating a token, then applies policy such as authorization checks, rate limits, and request validation. If the request passes, the gateway routes it to the appropriate back-end service, often over an internal network, and may add or strip headers, aggregate responses from several services, or cache results. It records logs and metrics throughout. Crucially, the gateway centralizes but does not replace service-level security: back-end services should still validate that requests come only from the gateway and enforce their own object-level authorization, because centralized checks at the edge cannot know which specific records a given user is allowed to touch. A well-run gateway is configured as code, kept patched, and monitored, since it is both the front door and a prime target.

Architecture Diagram



All client traffic enters through the gateway, which authenticates and applies policy before routing to internal services.

Visual Workflow



Common Attacks

- Bypassing the gateway to reach back-end services directly when they are exposed
- Exploiting a gateway misconfiguration to route or access unintended services
- Overwhelming the gateway itself to cause a broad denial of service
- Abusing overly permissive routing or transformation rules to reach hidden endpoints
- Harvesting secrets or keys stored insecurely in the gateway configuration

Common Mistakes

- Leaving back-end services reachable directly, so the gateway can be bypassed
- Treating the gateway as the only security layer and skipping service-level authorization
- Storing credentials or keys in the gateway config without proper secret management
- Enabling broad or wildcard routing rules that expose more than intended
- Failing to patch or monitor the gateway despite it being a prime target
- Terminating TLS at the edge but leaving internal traffic unprotected where required

Best Practices

- Make the gateway the only reachable entry point and block direct access to services
- Centralize authentication, rate limiting, and request validation at the gateway

- Keep object-level authorization in the services, since only they know the data context
- Manage gateway configuration as code, review changes, and keep it patched
- Store secrets in a dedicated secret manager, never in plain configuration
- Log and monitor all traffic and set alerts for abnormal patterns

Quick Checklist

- [x] Gateway is the only reachable entry point, direct service access blocked
- [x] Authentication and rate limiting enforced centrally at the gateway
- [x] Services still enforce their own object-level authorization
- [x] Routing rules scoped tightly, no unintended endpoints exposed
- [x] Secrets stored in a secret manager, not in configuration files
- [x] Gateway patched, configured as code, and monitored

Recommended Tools

API gateway
Centralizes authentication, rate limiting, routing, and logging for APIs

Identity provider
Issues and verifies the tokens the gateway validates

Secret manager
Stores keys and credentials the gateway needs securely

Observability platform
Collects gateway logs and metrics for monitoring and alerting

Industry Standards

OWASP API Security Top 10
Covers risks a gateway helps mitigate, such as broken authentication and unrestricted consumption

NIST SP 800-95
Guidance on securing web services, relevant to edge enforcement

NIST SP 800-53
Control families such as access control and boundary protection that a gateway supports

Career Relevance

API gateways are a core tool for backend engineers, API security engineers, and application security engineers who build and protect service architectures, and for platform and cloud engineers who operate them. Security engineers use gateways to enforce consistent edge policy and observe traffic. AI governance professionals also benefit from the concept, because gateways are a common place to apply access control and usage limits in front of AI services.

Interview Questions

- What problems does an API gateway solve in a microservices architecture?
- Why should object-level authorization stay in the services rather than only at the gateway?
- How do you prevent clients from bypassing the gateway to reach services directly?
- What are the risks of concentrating all traffic through a single gateway?
- How should secrets used by the gateway be stored and managed?

Related Certifications

OWASP-aligned application security training (vendor-neutral concepts)

Cloud and platform engineering certifications

CompTIA Security+ (foundational)

Further Reading

- OWASP API Security Project
- NIST Publications
- OWASP Cheat Sheet Series

Key Takeaways

- An API gateway is a single controlled entry point for API traffic.
- It centralizes authentication, rate limiting, routing, and logging in one place.
- Services must still enforce their own object-level authorization behind it.
- Block direct access so the gateway cannot be bypassed.
- The gateway concentrates risk, so harden it, patch it, and monitor it closely.

FAQ

▶ **Does an API gateway replace security in each service?**

▶ **What is the difference between an API gateway and a load balancer?**

▶ **Can an attacker bypass the gateway?**

Related Careers

RELATED ROLES

Application Security Engineer

Api Security Engineer

Backend Engineer

Security Engineer

RELATED CERTIFICATIONS

OWASP-aligned application security training

Cloud and platform engineering certifications (vendor-neutral concepts)

CompTIA Security+ (foundational)

CURRENT OPENINGS

Live roles are on the job board.

Browse all jobs

GUIDES & SALARY

Career guides

Role roadmaps

Salary data

Career tools

SUGGESTED LEARNING PATH

1. Ground the basics with CS-001 Cybersecurity
2. Study this sheet: **API Gateways**
3. Go deeper: API Rate Limiting
4. Go deeper: REST API Security
5. Validate it: work toward **OWASP-aligned application security training**
6. Find the role: browse current openings

RELATED SHEETS

CS-070 API Rate Limiting | CS-066 REST API Security | CS-068 OAuth 2.0 | CS-062 Authentication | CS-063 Authorization