What is rate limiting, and how do you choose the limits?

20 November 2026 · 4 min read · blogs.iictp.org/what-is-rate-limiting

Rate limiting caps how many requests a client can make in a period of time. It protects a service from overload, keeps one heavy user from crowding out everyone else, and slows down abuse such as password guessing and scraping.

TL;DR The short version

- Rate limiting caps how many requests a client can make in a period, protecting the service from overload and abuse.
- Limited clients should get HTTP 429 Too Many Requests, and limits should key on an API key, user or account before falling back to IP address.
- Set the numbers from measured capacity and real client behaviour, and enforce limits early, at the gateway or load balancer.

Read online, listen to it, or change the text size: https://blogs.iictp.org/what-is-rate-limiting

How do the common algorithms work?

A fixed window counts requests per minute or hour and resets at the boundary, which is simple but lets a burst through at the edge of two windows. A sliding window smooths that out. A token bucket adds tokens at a steady rate up to a maximum, and each request spends one, so it allows short bursts while holding the average rate.

What should the client see when it is limited?

In HTTP, the standard response is status 429 Too Many Requests, defined in RFC 6585. Include a Retry-After header saying when to try again, so well-behaved clients wait instead of retrying at once.

What do you limit by?

Limit by whatever identifies the party you are protecting against: an API key, a signed-in user, an account or, as a last resort, an IP address. IP addresses are shared by many users behind one network and are easy for an attacker to rotate, so they make a poor only key.

How do you choose the numbers?

Start from what the service can actually handle, measured under load, and from how real clients behave, taken from your logs. Set limits a little above normal use, alert when clients hit them, and review them as usage changes. Sensitive actions such as sign-in and password reset need much lower limits than reads.

Where should rate limiting run?

Run it as early as possible, at the API gateway or load balancer, so rejected requests cost little. In a service with many instances, keep the counters in a shared store, or each instance will allow the full limit on its own.

Related credential: Certified Technical Practitioner. Explore the exam facts and syllabus.

Sources and further reading

1. IETF. RFC 6585, Additional HTTP Status Codes. 2012.
2. IETF. RFC 9110, HTTP Semantics (Retry-After). 2022.
3. OWASP. API Security Top 10: Unrestricted Resource Consumption. owasp.org.

Published by IICTP, an independent technical credential body, on 20 November 2026. The current version of this post is at https://blogs.iictp.org/what-is-rate-limiting. More guides: blogs.iictp.org.