Engineer

Proxy

a server that forwards a request on your behalf

What it is

A proxy is a program placed between a client and the server it wants to reach. It accepts the client's request, makes a request of its own to the next machine along, and hands the answer back. The client talks to the proxy and the proxy talks to the server, so the two ends never share a connection. Proxies come in several kinds, and the useful question to ask of each one is who put it there.

HTTP's own specification, RFC 9110, names three common intermediaries. A proxy in the narrow sense is one the client chose, usually through local configuration such as a browser setting or an environment variable. A gateway, which the RFC also calls a reverse proxy, sits on the server's side: to whoever connects it behaves as the origin server, while it forwards each request inward to one or more servers behind it. A tunnel relays bytes blindly between two connections and stops being a party to the conversation once it is open. A client asks for one with the CONNECT method, naming a host and a port, and then runs TLS through it end to end. The RFC adds a fourth kind that nobody chose: the interception proxy, often called a transparent proxy, which redirects outgoing traffic on port 80 at the edge of a network, as public access points and corporate firewalls do.

Each hop is a separate connection. The RFC's diagram puts three intermediaries between a browser and an origin server, which makes four connections, and settings such as keep-alive or a timeout apply to one of those connections at a time. HTTP also asks a proxy to announce itself: it must add a Via header field to each message it forwards, which is how a loop or an unexpected intermediary shows up in a trace.

HTTP is only one layer. A SOCKS proxy, whose version 5 is specified in RFC 1928 from March 1996, relays TCP connections and UDP datagrams without reading what is inside them, so any protocol can use it. Tor is a familiar example: by default it listens on port 9050 for SOCKS connections from local applications.

Use the concept in a decision

Put “What is a proxy?” to work in a real decision. We can help relate the concept to the system you are building or buying. Wi-Fi, firewalls and remote access set up so staff can work from anywhere and only the right people get in.

Explore our services Get in touch →

Test the assumptions behind the term

Use the explanation and linked work to frame questions for your project. We can examine the assumptions with you before a tool is chosen. Managed 30 Ubuntu Linux VPS instances — 30 Linux servers managed, with recovery planned for

Managed 30 Ubuntu Linux VPS instances Get in touch →

Why it matters

Whoever runs the proxy can read the traffic. For plain HTTP that means every header and every body. Through a CONNECT tunnel it sees the host name and port, and the rest is TLS running from the client to the server. A reverse proxy that terminates TLS sees everything in clear, and that is its job, because it has to read a request to route it. So putting another company’s reverse proxy in front of a site, which is what a content delivery network does, hands that company each visitor’s address and each URL before the site itself sees them. That can be the right trade for a shop under attack. It is still a decision about privacy, and the privacy page should say so.

The second consequence is what the server learns. RFC 7239 describes the loss plainly: behind a proxy, requests appear to come from the proxy’s address. A log, a rate limit, a ban list or an allow-list keyed on the client address then sees one address for everybody. The usual repair is a header that carries the original address forward: the standard Forwarded field from RFC 7239, or the older X-Forwarded-For. Which copy of that header to believe is where most of the trouble starts.

Third, a proxy changes what an error means. A 502 Bad Gateway or a 504 Gateway Timeout is written by an intermediary about the server behind it: the proxy got an invalid response, or none in time. A 407 means the proxy itself wants credentials before it forwards anything. Knowing which machine wrote the error is half of the diagnosis.

Fourth, many proxies cache. RFC 9111 calls a cache that stores responses for more than one user a shared cache, usually part of an intermediary. It makes a site faster and cheaper to serve, and it is also the one place where a response meant for one person can reach another.

What it looks like in practice

In August 2026 the company moved its DNS zone of about twenty records into the repository and reconciled it against the Cloudflare API. Sending the web traffic through the provider's CDN as a reverse proxy was part of that work, and then a working CDN proxy in front of the site was dismantled for privacy. The objection was to where it sits: every reader's address and every requested path would reach a third party first, and for a company that tells its readers nobody is watching, that weighed more than the attack protection it bought.

Counting readers from the access log the web server already keeps, the following month, meant dealing with proxies the company does not run. The access log discards seven headers that relay a client's address before it writes a line. The log encoder copies headers unchanged, so without that step a reader behind a company proxy would have had an unmasked address stored under a header name the masking rule did not cover.

When the site gained an onion mirror on Tor that same month, the Tor daemon's own proxy port for outgoing traffic was disabled in its configuration. The onion service is reached through circuits the daemon establishes outward, so publishing this mirror requires no separate inbound Internet port. With its SOCKS listener disabled, this Tor daemon does not offer local applications an outbound proxy.

Where it goes wrong

Every request in the log comes from the same address. The application sits behind a reverse proxy and records the address of the connection it received, which is the proxy's. A rate limit then throttles everybody at once, and a ban written by an intrusion tool bans the proxy, which is to say the whole audience. Have the proxy pass the client address in a header, and configure the application to read that header only on connections from the proxy's own address. In nginx that is the realip module's set_real_ip_from directive; most web frameworks have an equivalent trusted-proxies setting.

A client picks its own address by sending a header. The application reads X-Forwarded-For on every request, including requests that reached it directly and requests that arrived with a forged value already in the header. RFC 7239 is explicit that the field can be modified by any node on the way, the client included. Trust only the entries your own proxy added, counting from the right, and bind the application to loopback or a private network so that the proxy is the only way in.

The log you meant to anonymise keeps full addresses. The client address is masked in the field somebody checked, but the same address travels in Forwarded, X-Forwarded-For, X-Real-IP and similar headers, and a logger that writes the header map as received stores them in full. List every header that can carry an address, drop or mask each one at the moment the line is written, then send a test request and read the line on disk.

Pages return 502 or 504 while the backend's own log stays quiet. The error was written by the proxy. A 502 means it got an invalid response: the backend was down, listening on another port, or speaking plain HTTP where the proxy expected TLS, or the reverse. A 504 means the backend took longer than the proxy's read timeout, which nginx sets to 60 seconds by default. Read the proxy's error log first, since it names the upstream address it tried, and raise a timeout only for the routes known to be slow.

One user is shown another user's page. A shared cache in a proxy or a content network stored a response that was personal, such as an account page returned alongside a session cookie, and served it to the next person who asked for the same URL. Mark every per-user response with Cache-Control: private, which RFC 9111 says a shared cache must not store, or with no-store where no copy should be kept anywhere, and check the headers the proxy actually returns.

Further reading

- RFC 9110, section 3.7: Intermediaries — the definitions of proxy, gateway, tunnel and interception proxy.
- RFC 9110, section 9.3.6: CONNECT — how a client asks a proxy to open a tunnel to a host and port.
- RFC 7239: Forwarded HTTP Extension — the standard header for the original client address, with its caveats.
- RFC 9111: HTTP Caching — shared and private caches, and the directives that keep them apart.

- RFC 1928: SOCKS Protocol Version 5 — the protocol-neutral proxy that relays TCP connections and UDP.

Glossary

Choose for your actual constraints

Bring the deployment path, the operational risks and the people on call. We can scope a change with checks, a recovery route and clear ownership.

See what it costs Get in touch →

Make the terminology practical

Ask how “What is a proxy?” applies to your organisation. A concrete example from your current system is enough to start the discussion.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/glossary/proxy/