Engineer

Network server

a computer or program that serves others over a network

What it is

A network server is a program that waits for other programs to reach it across a network and then answers what they ask. In everyday speech the word is shortened to *server*, and it names both that program and the computer it runs on. The program at the other end of each exchange is the client: a browser, a phone app, a script, a mail program, or another server asking a question of its own.

The standards behind the web put the role first. RFC 9110, which defines HTTP, calls an HTTP server “a program that accepts connections in order to service HTTP requests by sending HTTP responses”, and adds that the words client and server describe only the role a program plays on a particular connection. The same program can serve on one connection and ask on the next. A web server that passes a request on to an application behind it is that application’s client for the second hop.

Underneath most servers is one move, which TCP calls a passive open. The program tells the operating system that it wants to accept incoming connections, either on one local address or on any of them, and then it waits. A client makes the active open towards that address and port. While nobody is connected, a server does almost nothing visible.

Servers differ in how they wait. Many run for months at a time, holding their memory whether anyone visits or not. Others are started fresh for every connection: a supervising program does the waiting, launches a copy of the server for each visitor, and the copy exits when the visitor leaves.

Then there is the machine. NIST's glossary describes a server as "a computer or device on a network that manages network resources", and names file servers, print servers, network servers and database servers as examples. When a company says its site runs on one small server, it means a computer set aside to run server programs, today usually a virtual machine rented from a provider. This glossary keeps a firm line between the two words. A server is what answers. The machine, as a thing with a name and an address that is built, secured and counted, is a host. One host can run many servers, and a server can move to another host as long as its name moves with it.

The kinds most people meet are named for what they serve: web servers for HTTP, mail servers for SMTP and IMAP, DNS servers for names, database servers for queries, file servers, git servers and game servers. An application server usually sits behind a web server and runs the product's own code.

Use the concept in a decision

Put "What is a network server?" to work in a real decision. We can help relate the concept to the system you are building or buying. Day-to-day care of servers, mail, accounts and updates, so the things everybody relies on stay boring.

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. Ran 10 PostgreSQL and MS SQL servers on Ubuntu — 10 database servers deployed and kept healthy

Ran 10 PostgreSQL and MS SQL servers on Ubuntu Get in touch →

Why it matters

Everything a visitor reaches on the internet is answered by some server, so choosing servers is choosing what is exposed, what it costs and what fails together.

The first use of the word in a budget conversation is to ask which meaning is on the table. "We need a server" can mean a program to install, a machine to rent, a managed service that somebody else runs, or more capacity on a machine already paid for. Those differ in price, in who applies the security updates and in who is woken at night when one of them stops.

The second is cost, and most of a server's cost is its waiting. A program that holds memory while idle holds it around the clock, so on a small machine each resident server is a line in the budget. A server started per connection trades a little time on every visit for nothing held in between.

The third is exposure. Each listening server accepts input from strangers, which makes it something to keep patched, configure on purpose and watch. Running the fewest servers that do the job is a security decision as much as a saving.

The fourth is that the role can be bought instead of run. A managed database, a hosted mailbox or a content delivery network is a server that another company operates. The trade is control and visibility in exchange for fewer things to patch, and a small company can sensibly run its own web server while renting its mail.

What it looks like in practice

All of this site's servers fit on one machine, and the credits page names every server program on it. Caddy answers every web request and renews the HTTPS certificate itself. Agate and Gophernicus answer Gemini and Gopher, Tor answers the onion address, and Soft Serve is the git server. The four public copies of the site are served from one shared folder, and a single deploy brings every copy up to date at once. The onion service shows that serving and listening are separate things: it is the only front door that could be added without opening a port, because tor dials out to introduction points, and the machine's open ports stayed exactly as they were. Mail goes to a hosted service, which is the role bought rather than run.

The privacy page shows the per-connection kind from an angle few people look at. The Gopher copy is started afresh for each visitor by the operating system, which names that moment after the address the visitor came from and writes the name into a log of its own, one the server software never sees. The site found that log the day it first counted requests; it now expires after thirty days.

And one machine sets a limit on which servers it can take. The account of running the company's public services on one 512 MB host keeps mail off that box on purpose. The mail server exists as an install scaffold waiting for a host of its own, because a mail server needs headroom the machine has already spent: a limit written down rather than discovered later.

Where it goes wrong

The machine answers and the server does not. A ping, an open SSH session or a green tile for the virtual machine all say the host is up. None of them says that the web server is answering, that it returns the right page or that its certificate is

still valid. Check a server the way a client uses it, by asking a real question and reading the answer:

```
curl -sS -o /dev/null -w '%{http_code} %{time_total}s\n' https://example.com/
```

A 200 in a sensible time is what a visitor needs. Anything else is a fault, whatever the machine's graphs say.

The file on disk says one thing and the running server another. Most servers read their configuration when they start or reload and keep it in memory, so an edited file changes nothing until the next reload. A restart with a broken file leaves no server running at all. Validate the file, then hand it to the running server, then ask the server a question. Caddy's command line covers the first two steps: `validate` loads and provisions every module as if to start the configuration without starting it, and `reload` gives the running instance the new one.

```
caddy validate --config /etc/caddy/Caddyfile
caddy reload --config /etc/caddy/Caddyfile
```

Two servers do not halve the work of one. A second copy of an application server behind a load balancer calls every piece of state on the first into question. Sessions kept in memory belong to one copy, so a visitor sent to the other is logged out. A file uploaded to one disk is absent from the other. A scheduled job installed on both runs twice. Move sessions, uploads and schedules somewhere both copies share before adding the second, or keep one server and make it faster.

Every server on one machine shares its fate. One host carrying many servers is cheap and simple, and it also means that one reboot, one full disk or one exhausted memory takes every server down at once. For a small site that is a fair trade when it is chosen on purpose: the machine is described as code so it can be rebuilt, the data is backed up off the box, and a server that needs headroom of its own gets a machine of its own.

Further reading

- server — NIST CSRC Glossary — the definition NIST's publications share, with the kinds of server they name.
- RFC 9110, HTTP Semantics: Connections, Clients, and Servers — client and server defined as roles a program plays on one connection.
- RFC 9293, Transmission Control Protocol — passive and active opens, and how a listening program waits on one address or on all of them.
- Caddy: Command Line — `validate`, `reload` and the other commands of the web server this site runs.

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 network server?" 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/server/