Engineer

Ubuntu

a Linux distribution common on servers

What it is

Ubuntu is a Linux distribution published by Canonical: the Linux kernel and the thousands of programs around it, built, tested and patched by one publisher on a calendar it announces in advance. It is derived from Debian, so it uses the same `.deb` packages and the same `apt` tool to install them, and it adds its own release rhythm, its own defaults and a company that sells support. The same archive feeds desktops, cloud images and bare-metal servers; this entry is about the server.

Releases are named after the year and month they shipped, so 24.04 came out in April 2024. A new one appears every six months, and every second year the April release is a long-term support release, marked LTS. The difference is the length of the promise. An interim release gets nine months of fixes. An LTS release gets five years of standard security maintenance, which an Ubuntu Pro subscription extends to ten through Expanded Security Maintenance, and a Legacy add-on to fifteen.

On servers Ubuntu is usually weighed against two neighbours. Debian is its parent: the same packaging, with a community project rather than a company behind it. Red Hat Enterprise Linux and its rebuilds, such as AlmaLinux and Rocky Linux, form the other family, with their own package format and their own tools. Choosing among the three is mostly choosing a calendar, a support contract and the habits the team already has.

Use the concept in a decision

Put “What is Ubuntu?” 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

Once a team has picked Ubuntu, the decision that remains is which Ubuntu, and that decision sets the maintenance budget for years.

Servers belong on LTS releases. An interim release expires nine months after it ships, so a server that tracks them has to go through a release upgrade twice a year to stay patched, which is a lot of change for the privilege of slightly newer packages. Interim releases are where you try what the next LTS will bring.

An LTS release has a firm upgrade path, and it is narrower than people expect. A server moves from one LTS to the next one only, without skipping any: from 20.04 the way to 24.04 runs through 22.04. The release tool, `do-release-upgrade`, offers the jump only after the new release's first point release, so 24.04 became the default upgrade target once 24.04.1 was out. A fleet that falls two releases behind therefore needs two full upgrades per machine, each with its own testing.

Ubuntu Pro is the other half of the decision. It adds the five extra years of security fixes, Livepatch for applying kernel fixes without a reboot, Landscape for managing machines as a fleet, and FIPS 140-3 and CIS hardening tooling for regulated work. It is free on up to five machines, which covers a small business or a lab; beyond that it is a paid subscription. The honest comparison is the subscription against the cost of upgrading on schedule. Five years is enough for a server that gets rebuilt from code; ten years suits the appliance nobody wants to touch, where paying for patches beats migrating in a hurry.

Finally, Ubuntu has its own way of doing ordinary things, and a team arriving from another distribution should budget time to learn them. Network interfaces are configured in YAML through Netplan. The firewall is usually managed through `ufw`, a front end that ships disabled. Security updates install themselves every day through `unattended-upgrades`, which is on by default. Each of these is pleasant once known and surprising before.

What it looks like in practice

A fleet of Ubuntu Linux VPS instances that had grown one machine at a time is the clearest picture of what an unmanaged estate turns into. The configurations had drifted apart, and nobody had designed how the machines would be recovered or how they were networked. The work brought all thirty instances under management as one fleet: backups went in with restore procedures that were actually tested, and every instance's network configuration was moved onto the same hardened baseline. Recovery paths were now tested rather than assumed, and the downtime risk dropped.

Databases ran on the same footing elsewhere. Ten servers running PostgreSQL 14 and MS SQL Server 2019 on Ubuntu 20.04 LTS VPS instances were designed, deployed and maintained as one estate. Backups ran through `pg_dump` and SQL Server Agent jobs, with retention policies and offsite storage. Prometheus and Grafana tracked CPU, memory, disk I/O and query metrics. Both the databases and the Ubuntu OS were patched on a regular schedule, the firewalls were configured with UFW, and access was enforced with role-based access control. Disaster recovery was documented down to point-in-time restores and failover. The ten servers ran at 99.9% uptime, query response times improved by 30%, automation cut manual maintenance by 50% and freed more than ten hours a month, and the estate absorbed a 40% rise in user traffic without degrading.

Web hosting is the plainest case. Forty websites ran on Ubuntu Linux hosting servers behind a mix of Apache and Nginx, and the job was the configuration, the performance tuning and the maintenance that keeps sites reliable under real traffic. All forty ran with high availability and good performance.

Where it goes wrong

The updater installs a kernel fix, and the server goes on running the old kernel. `unattended-upgrades` installs security updates once a day, but its `Automatic-Reboot` option defaults to false, so a new kernel waits on disk until someone restarts the machine, and services keep using the old copies of updated libraries until they restart too. The file `/var/run/reboot-required` appears when a reboot is pending. Decide on a policy instead of leaving it to chance: alert on that file, schedule reboots in a quiet hour, or use Livepatch for the kernel. A scheduled reboot is two lines in `/etc/apt/apt.conf.d/50unattended-upgrades`:

```
Unattended-Upgrade::Automatic-Reboot "true";
Unattended-Upgrade::Automatic-Reboot-Time "03:30";
```

A server on an old LTS release stops getting security fixes while everything on it keeps working. Standard support for 20.04, for example, ended in May 2025; the machine boots, serves and reports green exactly as before. Record each server's release next to its end-of-support date, and before that date either attach Ubuntu Pro, which carries 20.04 to May 2030, or upgrade one LTS at a time.

A new `ufw` rule goes in, and traffic flows exactly as before. `ufw` evaluates rules in order and the first match wins, and a plain `ufw allow` or `ufw deny` lands after the rules already there. An older, broader rule above it decides first. List the rules with their positions and put the specific rule where it belongs:

```
sudo ufw status numbered
sudo ufw insert 1 deny from 203.0.113.7
```

A package from a third-party repository stays on an old, vulnerable version while the rest of the system is current. `unattended-upgrades` installs only from the origins listed in its `Allowed-Origins` setting, and out of the box that list holds only official Ubuntu repositories. A vendor repository or a PPA added later is updated by hand or not at all. Add its origin to the list deliberately, or put that package on a patch schedule of its own.

Further reading

- Ubuntu Server documentation — Canonical's own guides for installing, configuring and running Ubuntu on servers.
- Ubuntu release cycle — the six-month cadence, the LTS calendar and the end date of every release.
- Automatic updates — how `unattended-upgrades` chooses origins, and the reboot options that decide when fixes take effect.
- Firewall — `ufw` from first enable to numbered rules, with the netfilter layer underneath.
- How to upgrade your Ubuntu release — `do-release-upgrade`, the one-LTS-at-a-time rule and the point-release wait.

- Netplan documentation — the YAML network configuration Ubuntu servers use, including `netplan try` with its automatic rollback.

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 Ubuntu?” 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/ubuntu/