Engineer

Logging

writing down what happens, as it happens

What it is

Logging, also called event logging, is a program keeping a running account of its own behaviour, one entry per event: a request served, a password refused, a retry, a crash. Each entry, a log line or log record, carries at least a time and a message, so that someone can later reconstruct what happened and in what order. The OpenTelemetry project defines a log as a timestamped text record, structured or unstructured, with optional metadata, and notes that of all the telemetry signals, logs have the biggest legacy.

That legacy has a standard. Syslog, the long-standing protocol for passing event messages to a collector, was standardised as RFC 5424 in March 2009, which obsoletes the older RFC 3164, and it gives every message one of eight severities, from 0 for Emergency to 7 for Debug. Most logging libraries carry a shorter ladder of the same idea: debug, info, warning, error. OpenTelemetry's log data model maps them all onto one number from 1 to 24, in six bands from TRACE to FATAL.

An unstructured line is free text: easy for a person to read, and something a machine can only take apart with a regular expression written for that exact wording. A structured line follows a fixed layout, so every entry has the same named fields with the same types and meanings. OpenTelemetry prefers structured logs in production and adds a warning that is easy to miss: a line encoded as JSON is not structured just because it parses; what makes it structured is a stable layout. Shown across several lines here, a structured entry is a single line on disk:

```
{
    "time": "2026-09-14T03:12:07.412Z",
    "severity": "WARN",
    "service": "checkout",
```

```
    "event": "payment.retry",
    "attempt": 3,
    "trace_id": "4bf92f3577b34da6a3ce929d0e0e4736"
}
```

Metrics and traces are the signals usually collected beside it. Metrics are numbers sampled over time, cheap to keep and good at saying how often. Traces follow one request through every service it touches. A log keeps the detail of single events, which makes it what someone reads after an alert to find out why. OpenTelemetry ties logs to traces by attaching the IDs of the active trace and span to each log record.

Use the concept in a decision

Put “What is logging?” to work in a real decision. We can help relate the concept to the system you are building or buying. Monitoring and alerts that speak up while a fault is still small, and the figures that show how often it stays that way.

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. Built the company's first observability dashboard — the company's first view of its own systems

Built the company's first observability dashboard Get in touch →

Why it matters

A log is the only account of the past that most systems leave behind. A metric can say that errors rose at 03:12; the log says which request, from which service, with which value, and what the program did next. When something breaks at night and has recovered by morning, the log is the witness still there.

That account has a price, paid three ways. Every line costs disk, input and output, and processor time, and more again if it is shipped to a central store, where a hosted service typically bills by the volume taken in. Every line is also data held about someone: a web server’s access log holds the addresses and browsers of everyone who called. And every line has to be guarded. The OWASP Logging Cheat Sheet asks that each event record when, where, who and what, and that the logs themselves be protected against tampering and against unauthorised access, modification and deletion, because they may contain personal and other sensitive information.

At the scale of an organisation, NIST Special Publication 800-92, the Guide to Computer Security Log Management by Karen Kent and Murugiah Souppaya, published in September 2006, treats log management as infrastructure and processes run across the whole enterprise rather than a setting in one program.

Logs also work best next to numbers. The Prometheus project's instrumentation guide suggests a counter for every line of logging code, so that when an interesting message turns up, it is possible to tell how often it has occurred and over how long a period, and suggests comparing the totals of info, warning and error lines from one release to the next.

What it looks like in practice

The site's own access log is described line by line on the privacy page. Each request records the page and time, the response status, the browser's self-description, the name of the referring site and a blurred network address: the last part is replaced with zeroes at the moment the line is written, before anything touches the disk. A line lives about thirty days. The server software closes each file once a day, or sooner at about 10 MB, and deletes a closed file twenty-eight days later, and every night a check reads every file back and fails on any line older than thirty days. It has gone wrong twice, and the page records both. In September 2026 the quiet addresses filled 10 MB so slowly that their files had never been closed and still held every line since logging began, so a daily job was added to close them. On 26 October 2026 seven of the thirteen addresses that keep a log were still writing into files that job had marked as closed, and one kept a closed file forty-seven days old. The job is gone, and the server software now closes and deletes the files itself.

A full test run of a macOS application recorded 610,996 core log lines. A burst of core log events was retained in an unbounded main stream and copied to subscriber streams whose consumers did not use them. The incident record separately documents diagnostic reports of 111 GB of compressed pages on a 36 GB machine and a footprint sample showing growth of about 41 MB per second during a connection retry loop. Bounded buffers, subscriptions filtered by event type and a logging rate budget addressed that event path. The recorded full test-run comparison reduced core log lines to 1,411. It is not a production benchmark or a measured post-fix memory figure. The budget discards information by design: repetitive diagnostic events can be dropped and affected state refreshed instead of retaining unlimited events. Separate shutdown timeouts required their own correction.

On a production host, a hardening pass over thirteen systemd units turned up one directive misspelled in three unit templates for as long as those roles had existed. The service manager answers a key it does not recognise by logging that fact and starting the unit anyway, so the only trace of the mistake was a warning in a log; it came to light while the units were being rewritten. Now every unit goes through the service manager's own verifier during each converge, so the same typo fails a run.

Where it goes wrong

The log grows fastest when the system is already failing. A loop that logs each retry writes a handful of lines on a good day and thousands a second on a bad one, taking disk, memory and processor time from a system with none to spare. OWASP asks teams to verify that logging cannot be turned into a way to exhaust system resources, for example by filling the disk. Give every hot path a budget, collapse repeated messages into one line with a count, and keep a counter beside the message so the rate survives when lines are dropped.

Secrets and personal data get written down. A request logged whole carries whatever was in it. OWASP's list of what should usually be removed, masked, hashed or encrypted before it reaches a log includes session identifiers, access tokens, passwords, database connection strings, encryption keys and payment card data. Do it when the line is written, as the access log above does with addresses, because by the time anyone cleans up, the line may sit in a backup or an archive.

Input from outside writes a fake line. A user-supplied value containing a carriage return or newline, written unescaped into a text log, can start a new line that looks exactly like one the application wrote. OWASP calls this log injection and asks that such characters be sanitised and that data be encoded for the format it is written in. A structured logger that writes each value as an escaped JSON string does this by construction.

A log that goes quiet passes for a healthy system. A full disk, a rotated file the program never reopened or a collector that went down all look the same from outside: the lines stop arriving. OWASP asks for a way to detect that logging has stopped. Count lines per minute for each source and alert when a count drops to zero, so a gap in the log becomes an alert of its own.

Further reading

- Logs — OpenTelemetry on what a log is, why structured logs are preferred and how logs are tied to traces.
- Logs Data Model — the fields of a log record, its two timestamps and the severity scale from 1 to 24.
- RFC 5424: The Syslog Protocol — the syslog message format and its eight severities.
- NIST SP 800-92 — the Guide to Computer Security Log Management.
- Logging Cheat Sheet — OWASP on which events to record, which data to keep out and how to protect the result.
- Instrumentation — the Prometheus project's practices for instrumenting code, logging included.

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 logging?" 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/logging/