Engineer

Incident

an event that puts a system's data or service at risk

What it is

An incident is a problem in a running system that has become serious enough for people to drop their planned work and deal with it now: a service that has stopped answering, records that have been altered, a machine that has run out of memory, an account someone else is using. Security teams say “security incident”, operations teams say “production incident” or simply “outage”, and most organisations sort their incidents into severity levels.

NIST’s incident-response guidance draws the line in three steps, and they hold well outside security. An event is any observable occurrence involving computing assets: a login attempt, a software update, an app answering a request. An adverse event is one with a negative consequence, whatever caused it, and the guide puts natural disasters and power failures beside attacks. An incident is an adverse event that someone has judged to meet the organisation’s criteria, and the guide’s verb for that judgement is *declared*. The definition it relies on, taken from US federal law, covers an occurrence that actually or imminently puts information or a system at risk of losing its confidentiality, integrity or availability, or that breaks security policy.

Security and operations use the word with different centres of gravity. In security work an incident usually implies an adversary, a policy breach or data that may have left the building. In operations it means any unplanned interruption or serious degradation of a service, with no attacker in the story. The two overlap more than the vocabulary suggests: a denial-of-service attack and an ordinary traffic spike can look alike on a graph at first, and a disk that fills up can lose data as thoroughly as an intruder.

The neighbouring words are worth keeping apart. A defect is a flaw in the product; it can sit unnoticed for years and becomes part of an incident only when it starts doing harm. An alert is a message saying that something may be wrong; plenty of alerts never become incidents, and plenty of incidents are first reported by a customer. Service-management practice tracks the cause behind one or more incidents separately, as a problem.

Use the concept in a decision

Put “What is an incident?” 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

The decision hiding inside the word is a threshold. At what point does a slow page, an odd login or a growing queue stop being one engineer’s ticket and become the team’s first priority? Set the bar too high and a spreading breach gets handled as routine work by someone who cannot see it growing. Set it too low and people are pulled out of their work so often that declaring stops meaning anything. NIST’s answer is to agree the criteria before they are needed: incidents are declared when adverse events meet criteria the organisation has already defined, with known false positives taken into account. Written criteria with examples are far cheaper than an argument at two in the morning.

A declaration changes who does what. One person leads and coordinates rather than debugging, others are released from their normal work, one channel carries the facts and a timestamped record starts. Without that, the same people do the same work in parallel, and the first clear statement about the problem is the one a customer writes.

For some companies there is also a legal clock. The EU’s NIS2 directive, Directive (EU) 2022/2555, requires the organisations it covers to report a significant incident to the national authority: an early warning within 24 hours of becoming aware of it, an incident notification within 72 hours, and a final report within one month of that notification. Its definition reaches beyond attacks to any event compromising the availability, authenticity, integrity or confidentiality of data or

of the services a network and information system offers, so a severe outage can count. A team that starts asking “is this an incident?” on the second day has already spent the first window.

The running cost is people’s time, and it is larger than it used to be. The 2025 revision of NIST’s guide notes that incidents now happen frequently and that recovering from them often takes weeks or months, where its 2012 predecessor assumed they were rare and over within a day or two. Budgeting for incidents means budgeting for the follow-up work as well as for the night itself.

What it looks like in practice

One incident on this site concerns a macOS desktop application. Its diagnostic reports recorded 111 GB of compressed pages on a 36 GB machine on 2026-09-01; a separate footprint sample recorded growth of about 41 MB per second during a connection retry loop emitting core events. The response found log events retained in an unbounded main stream and copied to subscribers that did not use them. Buffers were bounded, subscriptions filtered by event type and logging given a rate budget. A full test-run comparison recorded 610,996 core log lines falling to 1,411. Those measurements concern the test run, not production traffic or post-fix memory. Architecture checks and application budget tests defend the bounded design. Its overflow policy may drop repetitive diagnostics and refresh affected state; separate shutdown timeouts needed their own correction.

Other incidents are the repeating kind. Eight WireGuard and IPsec tunnels between Google Cloud and Linux systems were fragile and slow to diagnose, and a dead tunnel could cut communication between environments. The work standardised their configuration, monitored their health and went after the recurring routing and key-exchange problems at the root rather than restarting the tunnels, and the recurring connectivity incidents stopped. When the same thing happens for the third time, the cause is the incident.

And some incidents begin with somebody trying to help. On a platform with no admin tool, approving a claim, hiding a review or clearing a cache falls to an engineer with a database console, which the project page calls slow, unlogged and one typo away from an incident. The operator back-office built there, around 40 admin routes made from 128 components, puts that work behind the same authorization and the same audit trail as the rest of the product, at the ongoing cost of a large internal surface to keep tested and accessible.

Where it goes wrong

Five people work on it and nobody leads it. The chat channel fills with theories, two engineers roll back different things, and customers hear nothing for an hour, because nobody declared and so nobody was given the job of coordinating. Let anyone on the team declare, and make the first act of a declaration naming one lead who keeps the picture and the record.

The same incident returns every few weeks. The response ends when the service is back, and a restart counts as the fix. Treat restoring service and removing the cause as two tasks: close the first on the night, and give the second an owner and a date before anyone goes back to bed.

The tooling makes the incident bigger. Logging that writes a line per event produces a log as large as the fault, and alert rules send hundreds of notifications for one cause. Give logs and alerts a budget in advance, collapse repeats, and test the reporting path under the load of the failure it reports.

The repair causes the next incident. A hurried change typed by hand against production fixes one symptom, starts another problem, and leaves a hole in the timeline the review needs later. Keep a short list of pre-approved recovery actions that leave a trail, and record every change made during an incident.

The lessons arrive months late, or arrive as blame. A review held after weeks of recovery remembers little, and one that asks who made the mistake teaches people to hide the next one. NIST's revised guide says lessons learned should often be shared as soon as they are identified, and its recovery guidance ends with an after-action report on the incident, the actions taken and what was learned. Write down what made the mistake easy, and change that.

Further reading

- Incident — the definitions NIST publications use, each shown with the law or standard it comes from.
- SP 800-61 Rev. 3 — NIST's incident response recommendations of 2025, built around the CSF 2.0 Functions instead of a single response cycle.
- Cybersecurity Framework — the framework whose Govern, Identify, Protect, Detect, Respond and Recover Functions the incident guide maps onto.

- RFC 2350: Expectations for Computer Security Incident Response — the IETF best current practice for what an incident response team should publish about itself.

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 an incident?” 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/incident/