

EXECUTIVE PAPER · INFRASTRUCTURE

# Every Change Should Be Reversible

Most large outages start with a change someone chose to make. Boards and CTOs who require every change to reach a small slice first, and to undo itself on failure, cap the damage before it reaches customers.

AUTHOR

Chris Watkins

SERVICE

Infrastructure

FOR

CTO, Board · 12 min read

AT A GLANCE

**The problem.** The costliest outages of 2024 and 2025 were self-inflicted. A security vendor's content file, a cloud provider's quota policy, and a tenant setting at a global edge network each reached the whole fleet within minutes. One of them cost large US companies an estimated $5.4 billion. Nobody had the chance to stop any of them at one percent.

**Why it persists.** Companies gate code releases and wave configuration, data, and vendor updates straight through, because those feel small and urgent. Rollback is rarely rehearsed, so teams try to fix forward under pressure. Boards review uptime figures and seldom ask how a change reaches production.

**The recommendation.** Treat reversibility as a release requirement. Every change, whether code, configuration, data, or a vendor update, goes first to a small group, gets checked against health signals, and reverses itself if those signals degrade. Measure how often changes fail and how fast they come back out. Write the same expectations into vendor contracts.

# The Decision in Front of You

Engineers ship code, administrators edit settings, and vendors push updates to your laptops and servers every day. The decision is whether each of those changes can reach every customer at once, or whether it must earn its way out in stages with a working undo button.

The stakes sit in a few bad days. Uptime Institute's 2026 outage report found that 57% of operators said their most recent major outage cost more than $100,000, and one in five put the cost above $1 million [1]. In Uptime's 2025 analysis, IT and networking problems accounted for 23% of impactful outages in 2024, a rise Uptime tied to growing complexity and the change-management mistakes and misconfigurations that come with it [2].

Speed is also climbing. Google's 2025 DORA survey of nearly 5,000 technology professionals found that AI adoption now raises software delivery throughput and still has a negative relationship with delivery stability [3]. The report's authors warn that more change volume produces instability unless teams have strong automated testing and fast feedback loops [3].

# What's Happening

### The pattern behind the largest recent outages

Each of the following failures began with a deliberate change that spread everywhere before anyone could judge whether it was safe.

**CrowdStrike, July 2024.** On July 19, 2024, CrowdStrike released a content configuration update for its Falcon sensor on Windows [4]. A new template expected 21 input fields, the sensor code supplied 20, and the mismatch caused an out-of-bounds memory read that crashed the machine [4][5]. A logic error in the content validator let the file through, and this category of content had not faced the same tests and reviews as sensor software [5]. Microsoft estimated about 8.5 million Windows devices were affected, under 1% of all Windows machines [6].

The cost landed on CrowdStrike's customers. The insurer Parametrix estimated $5.4 billion in direct losses for the US Fortune 500 excluding Microsoft, with 125 companies affected, including all six airlines on the list and roughly three quarters of the healthcare and banking firms [7]. Parametrix put insured losses at only $0.54 billion to $1.08 billion [7]. Delta Air Lines told the SEC it expected about $550 million in losses, after canceling roughly 7,000 flights over five days [8].

CrowdStrike's fixes include deployment rings, canary testing before wider release, rollback when problems appear, and customer control over when content updates land [4][5].

**Google Cloud, June 2025.** In May 2025, Google rolled out new quota-checking code in Service Control, which authorizes API requests across Google Cloud [9]. The rollout went region by region, but the new code path ran only when a certain kind of policy changed, so the staged release never exercised it. The code lacked error handling and had no feature flag, a switch that turns new behavior on for a few customers at a time [9].

On June 12, 2025, a policy update with blank fields landed in a database that replicates globally within seconds, and Service Control crashed in a loop in every region [9]. Engineers found the cause in about 10 minutes. Most regions still took around two hours to recover, and one large US region took close to three as restarting services overloaded the database behind them [9]. Google's report says a feature flag would have caught the problem in staging, and its fixes require flags, off by default, on critical binaries [9].

**Microsoft Azure Front Door, October 2025.** On October 29, 2025, a tenant configuration change in Azure Front Door, Microsoft's global edge routing service, left many edge nodes unable to load [10]. Safeguards meant to block faulty deployments were bypassed because of a software defect, and at least 20 Azure services were affected, including the Azure Portal [10]. Microsoft's post-incident summary, as relayed by affected customers, puts the impact from 15:41 UTC on October 29 to 00:05 UTC on October 30, about 8 hours 24 minutes [11]. Recovery meant freezing configuration changes and pushing the last known good configuration worldwide [10].

**Cloudflare, December 2025.** On December 5, 2025, a configuration change made to shield customers from a new software vulnerability caused 28% of applications on Cloudflare's network to fail for about 25 minutes [12]. Its configuration system reached 90% of servers within seconds, while its software releases already moved through staged gates with automatic rollback [12]. In 2026 it finished applying the same staged, health-checked rollout and automated rollback to configuration changes [13].

**Exhibit 1. In each case a change reached the whole fleet before anyone could judge it**

| Incident | What changed | How far it spread | Impact | What would have contained it |
|---|---|---|---|---|
| CrowdStrike, July 2024 | Vendor content update to an endpoint agent | Every online Windows host running the sensor | About 8.5 million devices; est. $5.4B Fortune 500 losses | Canary ring before broad release; customer control over update timing |
| Google Cloud, June 2025 | Policy data read by unflagged new code | Every region within seconds | Most regions about 2 hours; one region nearly 3 | Feature flag off by default; incremental data propagation |
| Azure Front Door, October 2025 | Tenant configuration change | Global edge fleet | About 8 hours 24 minutes; 20+ Azure services | Validation that cannot be bypassed; staged config rollout; last known good fallback |
| Cloudflare, December 2025 | Security configuration change | 90% of servers within seconds | 28% of applications failed for about 25 minutes | Same staged, health-checked rollout already used for software |

Source: CrowdStrike [4], TechTarget coverage of the CrowdStrike RCA [5], Microsoft via TechCrunch [6], Parametrix [7], Google Cloud [9], The Stack [10], customer-posted summary of Microsoft's post-incident review [11], Cloudflare [12][13].

## What the measurements say

DORA's 2024 research grouped teams into performance clusters. The top group saw about 5% of changes fail and recovered from a failed deployment in under an hour. The bottom group saw about 40% fail and took between one week and one month to recover [14]. A team at the bottom fails eight times as often and takes far longer to undo each failure.

In Uptime's 2025 report, nearly 40% of organizations had suffered a major outage caused by human error in the prior three years, and 85% of those incidents came from staff not following procedures or from flawed procedures [2]. The share tied to skipped procedures rose ten points in 2025 [2]. A staged rollout with automatic rollback is a procedure the system enforces even when a tired person skips a step.

Regulators are already writing this down. The EU's Digital Operational Resilience Act technical standard on ICT risk, Commission Delegated Regulation (EU) 2024/1774, requires financial entities to plan and test changes in a controlled way, keep approval independent from the people requesting and implementing a change, and define fall-back procedures and responsibilities, including for aborting a change [15]. Emergency changes must be documented and re-assessed after the fact [15].

## Why the Usual Approach Fails

Most companies already have change control, usually a ticket, a weekly approval meeting, and a maintenance window. That process was built for large, infrequent software releases, and recent outages started elsewhere.

Configuration and data move faster than code. In three of the four cases above, the harmful change was a setting, a policy record, or a content file. Each traveled a fast path built to save minutes on good changes, with no stage where a bad one could be caught small [5][9][12].

Approval is a weak test of safety. A review meeting confirms that a change looks sensible on paper, and the defects that matter tend to appear only under live traffic. Google's own staged rollout passed the faulty code region by region because nothing triggered it [9]. Exposure to a small amount of real traffic, watched closely, is what catches them.

Rollback is assumed and seldom practiced. Google's site reliability engineers have advised since at least 2017 that rollback should be the first response to a suspected bad release, with diagnosis afterward, and that teams should roll back periodically just to prove they still can [16]. Untested rollback makes engineers hesitate, and the outage grows while they try to fix forward.

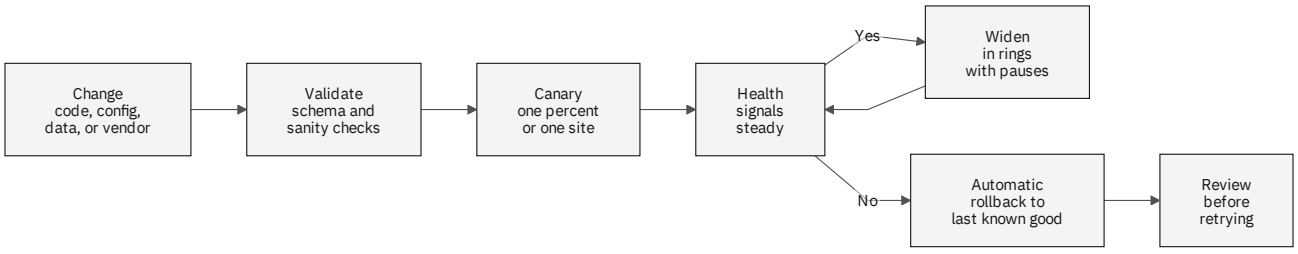
Vendor changes sit outside the process entirely. CrowdStrike's customers had no say over when the faulty content arrived on their machines [4]. Many companies still accept vendor auto-updates with no ring of their own and no contractual right to delay.

## The Framework: The Reversibility Gate

A safe change reaches a small, representative slice of production first, automated health checks compare that slice against the rest, and the change reverses itself if the checks degrade, without waiting for a human decision. Google's production guidance describes the same structure: nonemergency rollouts proceed in stages, each stage is supervised by monitoring, and unexpected behavior triggers rollback first and diagnosis second [17].

---

**Exhibit 2. The Reversibility Gate: no change reaches everyone until a small slice proves it safe**




Source: author's framework, consistent with Google SRE production guidance [17] and the corrective actions published by CrowdStrike [4], Google Cloud [9], and Cloudflare [13].

**Validate.** Reject malformed input before it ships. A file with an unexpected field count or blank values should fail in seconds. When a system receives implausible configuration, it should keep running on its previous state and raise an alert [17]. Cloudflare now calls this "fail stale" [13].

**Canary.** Send the change to a small slice first: one percent of traffic, one site, one internal team. Cloudflare starts with employee traffic, then free customers, before paying ones [12].

**Watch.** Compare the canary's error rates, latency, and crash counts against the baseline automatically, using signals defined before the change ships.

**Widen or reverse.** If signals hold, widen in rings with pauses between them. If they degrade, roll back automatically to the last known good version. A person reviews the failure before anyone retries.

The arithmetic of blast radius is simple. For a hypothetical distributor with 40 sites, a bad change pushed everywhere and fixed in four hours costs 160 site-hours of disruption. Sent to one site and reversed by health checks in 15 minutes, it costs a quarter of a site-hour, 640 times less. This is the author's illustrative estimate; it assumes the canary site is representative and the health checks detect the fault.

## Costs, Risks, and Tradeoffs

Reversibility costs engineering time and slows the fastest changes. Some changes cannot be reversed at all.

**Exhibit 3. Reversibility has real costs, and some changes need a different control**

| Area | Cost or risk | Mitigation |
|---|---|---|
| Release speed | A staged rollout adds minutes to hours before a change reaches everyone | Size stages to the risk; let low-risk changes move through rings faster |
| Urgent security updates | Waiting on rings delays protection against an active threat | Keep a short emergency path with a small canary and a fast ring sequence, documented and reviewed afterward as EU 2024/1774 expects [15] |
| Engineering investment | Feature flags, health checks, and automatic rollback take work to build into each system | Start with the five systems whose failure costs most per hour |
| Irreversible changes | Database schema changes, data deletions, and sent messages cannot be undone by redeploying | Split them into reversible steps, ship the code that reads both formats first, and keep backups verified before the irreversible step [16] |
| False confidence | A canary that is not representative passes a change that fails elsewhere | Rotate canary sites across regions and traffic patterns [17] |
| Vendor bargaining power | Smaller buyers may not win contract terms on update control | Use product settings for update rings where they exist; hold some hosts back from automatic updates |
| When this is wrong | A small company with one site, running mostly on SaaS, has little fleet to stage across and little ability to build rollback tooling | Rely on vendor practices and contracts, keep good backups, and schedule changes for low-traffic hours |

Source: author's analysis, with references as cited.

# How to Act

**Exhibit 4. A one-year roadmap to reversible change**

| Window | Actions | Owner | Milestone |
|---|---|---|---|
| First 30 days | List every path by which production changes: code pipelines, configuration tools, database scripts, and vendor auto-updates. Mark which paths have stages and which reach everything at once. | CTO with platform or operations lead | Change-path inventory complete; every all-at-once path named with an owner |
| First 30 days | Start measuring change failure rate and time to recover from a failed change for the top five revenue systems. | Engineering leads | Baseline figures reported to the CTO |
| First 90 days | Put the top five systems behind staged rollout with automated health checks and one-command rollback. Route configuration changes through the same pipeline as code. | Platform engineering lead | Five systems staged; configuration no longer bypasses the pipeline |
| First 90 days | Review contracts and product settings for critical vendors that push updates. Configure update rings where the product allows. | CIO with procurement and security | Ring settings applied; gaps listed for renewal negotiation |
| First 90 days | Rehearse a rollback on each of the five systems. | Site reliability or operations lead | Five successful rollback drills, timed and recorded |
| First 365 days | Extend staged rollout and automatic rollback to all customer-facing systems. Add feature flags to new work by default. | CTO | Change failure rate below 15% and median recovery from failed change under one hour on critical systems |
| First 365 days | Report change failure rate, recovery time, and any all-at-once changes to the board risk committee each quarter. | CTO with chief risk officer | Four quarterly reports delivered; board sees the trend |

Source: author's recommendation. Targets are illustrative; DORA's 2024 top cluster recorded about 5% change failure and under one hour to recover [14].

# Questions to Ask Your Team

1. Which of our production systems can change everywhere at once, and who can make that change?
2. Do configuration changes and database changes go through the same staged pipeline as code?
3. What share of our changes failed last quarter, and how long did each take to undo?
4. When did we last roll back a production change on purpose, and how long did it take?

5. Which vendors can push software or content to our servers and laptops without our approval?
6. For our five most important systems, which health signals would trigger an automatic rollback?
7. Which changes cannot be reversed, and what extra checks do they get?

## Closing

The outages that cost the most in recent years started with routine changes released to everyone at once. A change that reaches one percent of customers and reverses itself costs minutes. The same change at full scale cost Fortune 500 companies billions in a single day in 2024 [7]. Require every change, including configuration and vendor updates, to prove itself on a small slice and to undo itself when it fails, and ask to see the change failure rate alongside revenue.

# Sources

[1] Uptime Institute. "Uptime Announces Annual Outage Analysis Report 2026." Press release via Business Wire. 2026. https://www.businesswire.com/news/home/20260513289344/en/Uptime-Announces-Annual-Outage-Analysis-Report-2026.

[2] Uptime Institute. "Uptime Announces Annual Outage Analysis Report 2025." Press release via Business Wire. 2025. https://www.businesswire.com/news/home/20250506733553/en/Uptime-Announces-Annual-Outage-Analysis-Report-2025.

[3] Google Cloud. "Announcing the 2025 DORA Report." 2025. https://cloud.google.com/blog/products/ai-machine-learning/announcing-the-2025-dora-report?e=48754805.

[4] CrowdStrike. "Executive Summary: Root Cause Analysis, Channel File 291." 2024. https://crowdstrike.com/wp-content/uploads/2024/08/Executive-Summary_Root-Cause-Analysis_Channel-File-291.pdf.

[5] TechTarget. "CrowdStrike details errors that led to mass IT outage." Coverage of CrowdStrike's external technical root cause analysis. 2024. https://www.techtarget.com/cybersecurity/news/366602392/CrowdStrike-details-errors-that-led-to-mass-IT-outage?amp=1.

[6] TechCrunch. "Microsoft says 8.5M Windows devices were affected by CrowdStrike outage." 2024. https://techcrunch.com/2024/07/20/microsoft-says-8-5m-windows-devices-were-affected-by-crowdstrike-outage/.

[7] Parametrix. "CrowdStrike to Cost Fortune 500 $5.4 Billion; Insured Loss Range of $540 Million to $1.08 Billion." 2024. https://www.parametrixinsurance.com/in-the-news/crowdstrike-to-cost-fortune-500-5-4-billion-insured-loss-range-of-540-million-to-1-08-billion.

[8] Fox Business. "Delta estimates CrowdStrike outage cost airline $380M in revenue." Coverage of Delta Air Lines SEC filing. 2024. https://www-ak-ms.foxbusiness.com/lifestyle/delta-estimates-crowdstrike-outage-cost-airline-380m-revenue.

[9] Google Cloud. "Multiple GCP products are experiencing Service issues." Incident report, Google Cloud Service Health. 2025. https://status.cloud.google.com/incidents/ow5i3PPK96RduMcb1SsW.

[10] The Stack. "Azure network outage." Coverage of Microsoft statements on the Azure Front Door incident. 2025. https://www.thestack.technology/azure-network-outage/.

[11] Cambrionix. "Azure Front Door and CDN Global Service Disruption (Tracking ID: YKYN-BWZ)." Customer status page summarizing Microsoft's post-incident review. 2025. https://status.cambrionix.com//incidents/01K9SX68G1VRSQHZF0ZR8X5XXS.

[12] Cloudflare. "Code Orange: Fail Small. Our resilience plan following recent incidents." 2025. https://blog.cloudflare.com/fail-small-resilience-plan/.

[13] Cloudflare. "Code Orange: Fail Small is complete. The result is a stronger Cloudflare network." 2026. https://blog.cloudflare.com/code-orange-fail-small-complete/.

[14] Octopus Deploy. "The 2024 DevOps performance clusters." Summary of the 2024 Accelerate State of DevOps Report (DORA). 2024. https://octopus.com/blog/2024-devops-performance-clusters.

[15] European Commission. Commission Delegated Regulation (EU) 2024/1774, Article 17, "ICT change management." OJ L, 2024/1774. 2024. https://www.springlex.eu/en/packages/dora/rts-rmf-regulation/article-17/.

[16] Hilton, A. "Reliable releases and rollbacks: CRE life lessons." Google Cloud Blog. 2017. https://cloud.google.com/blog/products/gcp/reliable-releases-and-rollbacks-cre-life-lessons/.

[17] Beyer, B., Jones, C., Petoff, J., and Murphy, N. R., eds. *Site Reliability Engineering: How Google Runs Production Systems*, Appendix B, "A Collection of Best Practices for Production Services." O'Reilly. 2016. https://sre.google/sre-book/service-best-practices/.

---

The views in this paper are the author's own and do not represent any employer. Scenarios are hypothetical. This paper is not legal, financial, or compliance advice.