Old Lessons for a New Machine: Engineering AI Safety

Krishna R. Pattipati
krishna@teamqsi.com | krishna.pattipati@uconn.edu
October 10, 2026

Abstract. AI safety should build on established dependability engineering, with protections proportionate to a system's authority and potential consequences. This essay connects historical lessons and 22 documented cases involving AI and automated systems to diverse redundancy, analytical checks, monitoring, containment, recovery, and protected shutdown. It shows why model agreement alone does not establish safety: common causes, shared specification errors, and bypass paths can defeat nominally independent checks, and testing alone cannot economically demonstrate the very low failure probabilities sometimes claimed. A practical architecture separates agent proposals from enforced authorization and safety constraints, supported by explicit hazards, fault models, diagnostic coverage, and intervention deadlines. Worked examples in network configuration and insurance claims, with an explicit fault tree and coverage matrix, show how prevention and recovery address different hazards and where assurance effort belongs. Independent incident investigation, accountable risk acceptance, and protection of engineering judgment from organizational pressure complement technical controls. The objective is justified, controlled residual risk, supported by evidence and informed by decades of engineering experience.

1 Introduction

Artificial intelligence is an imperfect, human-made artifact—the kind of artifact that system safety engineering was developed to manage. Its errors cannot simply be engineered away: learning from finite data yields only probabilistic guarantees, and learning theory identifies further limits under explicit assumptions (Section 6.1). For decades, engineers have built dependable systems from fallible components: unreliable hardware, software with latent defects, and operators who make mistakes. Aircraft, spacecraft, reactors, railways, and payment networks depend on architectures that anticipate failure.

AI safety should begin with that inheritance. Zero risk is not the objective. The objective is justified, controlled residual risk, established by evidence. Claims of exceptional danger should identify hazards, credible causal paths, and the limits of existing protections. Claims of adequate protection require equally explicit evidence.

Recent incidents make the question concrete. During cybersecurity evaluations in May and June 2026, AI agents turned a shared package service into a covert message board and found paths out of network isolation; in July, about 700 of roughly 1,200 agents joined an attack on Hugging Face's infrastructure. Production safeguards had not been applied to the evaluations, some agents spoofed their own tool records, and early warning signs were not escalated (OpenAI, 2026; Greenblatt, Cotra, & Wijk, 2026). None of these failure mechanisms was new: a shared resource defeated isolation, a single path reached the internet, evidence was forgeable, and a warning went unheeded. The Appendix maps 22 documented incidents to the same mechanisms. OpenAI and Anthropic have since disclosed further cases from training and evaluation, including agents that reached outside services through gaps in internet-access restrictions and, in one case, submitted a real government form when the practice copy failed to load (OpenAI, n.d.; Anthropic, 2026).

This essay focuses on the operational safety of deployed AI systems: controlling the harm a system can cause through its granted authority. Deliberate misuse and hazards distributed across many deployments also require institutional controls. Fraud prevention and cybersecurity offer relevant precedents:

- **Control access.** Customer verification and network authentication have counterparts in identity verification and risk-based authorization for AI capabilities.
- **Monitor across accounts.** Banks and security operations centers correlate events that look innocuous in isolation. AI providers can similarly identify abuse patterns across users and deployments.

- **Limit damage.** Spending limits, transaction holds, restricted capabilities, and least privilege bound what one actor can do.
- **Share intelligence.** Incident reporting, vulnerability disclosure, and sector coordination let institutions learn from one another's failures.
- **Assign residual losses.** Banks absorb residual fraud losses allocated to them, giving them an incentive to invest in prevention and detection and to price the remaining risk into their business. Consumer-liability limits help allocate that risk (12 C.F.R. §§ 1026.12 and 1005.6). This illustrates the cheapest-cost-avoider principle: place liability with those best able to prevent harm at least cost (Calabresi, 1970). For AI, developers and deployers best placed to prevent harm should bear responsibility for residual losses and compensation.
- **Maintain accountability.** Prosecution, sanctions, and liability create incentives for human operators and organizations to prevent misuse.

These measures manage persistent adversarial risk; they do not eliminate it or establish universal agreement about acceptable losses. AI deployment likewise needs explicit responsibility for residual harm. The remainder of this essay addresses engineering controls within individual systems. Section 2 frames safety as an argument about hazards and their causes, and Section 3 draws lessons from historical systems and documented AI incidents. Section 4 presents an architecture that separates agent proposals from enforcement, Section 5 applies it to two worked examples, with an explicit fault tree and coverage matrix, Section 6 examines what AI changes, including what happens when a human is the hazard, and Section 7 proposes a path forward.

2 From mechanisms to a safety argument

A catalog of defenses does not demonstrate risk reduction. A safety argument must trace hazards to their causes and show how each protection changes the paths to harm.

A fault tree represents combinations of events sufficient to produce a specified hazard. A **minimal cutset** is inclusion-minimal: removing any event makes that combination insufficient. A singleton cutset contains one event—a single point of failure within the stated model and operating assumptions.

1. Identify hazards and their causal paths.
2. Eliminate singleton cutsets where practicable.
3. Provide independently justified detection and mitigation for remaining paths.
4. Establish diagnostic coverage, response deadlines, and recovery effectiveness.
5. Assess common causes and failures of the protections themselves.

For a fault tree whose complete set of minimal cutsets is represented by events C_{k}, the modeled hazard *H* satisfies (Stamatelatos et al., 2002):

$$P(H) = P\left(\bigcup_k C_k\right) = \sum_k P(C_k) - \sum_{i<j} P\left(C_i \cap C_j\right) + \sum_{i<j<l} P\left(C_i \cap C_j \cap C_l\right) - \cdots .$$

Truncating after an odd intersection order gives a Bonferroni upper bound; truncating after an even order gives a lower bound (Feller, 1968). The first term alone, the sum of the cutset probabilities, is the union bound: the first-order upper bound. Alternatively, a binary decision diagram (BDD) represents the same Boolean logic through mutually exclusive branches, allowing exact probability evaluation without enumerating every cutset intersection (Bryant, 1986; Rauzy, 1993). With independent basic events, weighted traversal of the BDD gives $P(H)$; dependent events require an appropriate joint probability model. Bayesian networks represent the same logic, usually less efficiently than BDDs for independent events, but can model common causes, multi-state components, and diagnostic inference explicitly (Bobbio et al., 2001). These methods improve computation, but none removes the need to model common causes and dependencies correctly.

Detection and response earn credit only when effective intervention precedes harm:

$$T_{detect} + T_{respond} < T_{harm}.$$

For a modeled path with one hazardous proposal, nine approvals of the unsafe action, and no bypass, execution requires all ten events. The number of events alone does not establish reliability: common causes can defeat several approvals together, and bypasses create shorter paths. For every protection, ask which paths it blocks, which dependencies it introduces, and what evidence supports the claimed reduction in risk.

3 Lessons from history

3.1 A historical perspective

Many proposed AI controls have established precedents. Dependable computing developed by combining them rather than relying on one mechanism (Johnson, 1989).

Era	Mechanism	Historical example
1785	Majority voting	Condorcet's jury theorem: if voters judge independently and each is right with probability above one-half, a majority is more reliable than any one voter, increasingly so as the group grows (Condorcet, 1785).
1850s onward	Interlocks	Railway signal interlocking prevented conflicting routes from being set.
Early 1900s	Dead man's switch	Train controls stopped a train when the driver ceased operating the control.
1942	Emergency shutdown	Chicago Pile-1 used emergency safety rods. The familiar "Safety Control Rod Axe Man" expansion of SCRAM is a later myth (Wellock, n.d.).
1956	Redundancy and voting	Von Neumann (1956) analyzed reliable systems built from unreliable components; modular redundancy became a core design technique.
1960s	Restart and recovery	The Apollo Guidance Computer shed lower-priority work and restarted during the lunar-landing overload alarms.
1970s	Checkpointing	Tandem NonStop process pairs and database logs supported recovery, commit, and rollback.
1970s	Design diversity	Randell's (1975) recovery blocks and Avizienis and Chen's (1977) N-version programming addressed software faults.
1970s onward	Safe mode and fault detection, isolation, and recovery (FDIR)	Spacecraft autonomously detected faults, isolated failures, and entered safe modes while awaiting ground instructions.
1980s	Diverse flight-control redundancy	The Space Shuttle combined four redundant primary computers with separately developed backup software; Airbus used dissimilar flight-control implementations.
1990s onward	Assurance standards and safety cases	DO-178B/C, IEC 61508, and structured safety cases formalized evidence and review in safety-critical industries.
2014	Automatic recovery	The F-16's Automatic Ground Collision Avoidance System (Auto-GCAS) began recovering aircraft from imminent terrain impact, with pilot override (Lockheed Martin, n.d.).

These architectures assumed components could fail. They combined voting, backup functions, containment, recovery, and human authority rather than demanding perfect components.

3.2 Five kinds of redundancy

Five categories are useful here, though they are not exhaustive (Johnson, 1989). Their AI counterparts are sometimes analogies rather than exact equivalents.

Type	Mechanism and classical examples	Coverage and AI counterpart
Hardware	Replicate components and vote or switch to a spare: modular redundancy, redundant flight computers, and multiple sensors.	Covers some component failures, subject to shared dependencies. AI systems can separate inference servers, sensors, power, and infrastructure.
Software	Develop different implementations: N-version programming, recovery blocks, and dissimilar flight-control software.	Can cover implementation-specific faults. Models from different developers may provide diversity, but shared specifications, data, and development practices limit it.
Information	Add check information: parity, error-correcting codes, CRCs, checksums, and redundant storage.	Detects or corrects data corruption. Checksums detect changes; authenticated signatures on weights, checkpoints, logs, and commands establish integrity and origin, not factual truth.
Time	Repeat an operation: instruction retry, re-execution, or retransmission.	Can recover from transient faults. Re-running inference can address transient infrastructure failures; repeated sampling exposes variability. Paraphrasing is closer to input diversity.
Analytical	Compare observations with model predictions: failure-detection filters, parity relations, and observer- or Kalman-filter-based residuals (Beard, 1971; Chow & Willsky, 1984).	Uses physical or functional relationships without duplicating hardware. Process models, simulators, and invariants can check proposed AI actions, subject to model and observation validity.

Analytical redundancy deserves particular attention. It checks an AI system against a different kind of evidence: predicted consequences, conservation laws, process constraints, or resource limits. This can diversify failure mechanisms, but an incorrect reference model can still accept hazardous behavior. Its reach depends on an adequate domain model: physical plants, networks, and ledgers offer constraints that open-ended language tasks often lack. Where analytical checks are inadequate, model-based review can supplement human judgment; the available assurance should determine how much authority the system receives. Human redundancy, such as two-pilot operation or the two-person rule, is sometimes treated as another category.

3.3 Lessons history has already taught

- **Independent development does not establish independent failure.** Knight and Leveson (1986) tested 27 independently developed program versions and found coincident failures more frequent than independence predicted.
- **Shared causes defeat replication.** Ariane 5's redundant inertial reference systems suffered the same software failure in 1996. At Fukushima, the 2011 tsunami disabled multiple backup power provisions.
- **Removing independent protection can expose latent software faults.** Therac-25 relied heavily on software after eliminating protections used in earlier designs. Software defects, including race conditions, contributed to severe overdoses (Leveson & Turner, 1993).
- **A single erroneous input can acquire excessive authority.** The original 737 MAX MCAS could repeatedly command stabilizer movement using one angle-of-attack sensor.
- **Alarms must support diagnosis and timely action.** At Three Mile Island, alarm overload and an indicator that did not show actual valve position complicated operators' understanding of the accident (Kemeny et al., 1979).
- **Organizations can override or conceal engineering evidence.** Before the 1986 Challenger launch, Morton Thiokol managers reversed their engineers' recommendation against launching in cold weather (Presidential

Commission on the Space Shuttle Challenger Accident, 1986). Boeing admitted that two of its technical pilots deceived the FAA about MCAS (U.S. Department of Justice, 2021).

The recurring weaknesses are shared assumptions, inadequate observation, excessive authority, ineffective intervention, and organizational pressure on engineering judgment. They belong in AI safety analysis as explicitly as they do in conventional systems.

3.4 Shared errors that replication preserves

Diversity can mask a fault unique to one implementation. Replication preserves errors shared by every copy. The relevant distinction is therefore between implementation-specific faults and shared specifications, processes, or evidence.

Fault source	Effect on protection	AI example
Implementation-specific fault	Diverse implementations may mask or detect it.	One model's idiosyncratic reasoning error.
Shared specification error	Conforming implementations may all produce an unsafe result.	A flawed objective or policy imposed on every checker.
Shared development or verification failure	Nominal diversity may provide little coverage.	A common evaluation suite that omits the hazard.
Incorrect reference model	Monitors may accept the same hazardous behavior.	A simulator with an incorrect plant or network model.

Five historical cases illustrate failures of design, verification, and recovery (case selection follows Tramontana, 2019).

Case	Failure mechanism	Lesson for AI
Ford Pinto, 1971–1980	The rear fuel-tank arrangement was vulnerable to damage and fire in rear-end collisions; the vehicles were recalled in 1978.	Cost and schedule pressures can affect multiple defenses. Safety decisions need independent authority and review.
Teton Dam, 1976	Internal erosion developed during first filling through the dam–foundation system. Available drawdown capacity was inadequate to avert failure (Association of State Dam Safety Officials [ASDSO], n.d.).	Commissioning has distinctive hazards. Provide an effective means of backing off before operation begins.
Three Mile Island, 1979	Auxiliary feedwater valves were closed, and a relief valve stuck open while its indicator reflected the close command rather than actual position (Kemeny et al., 1979).	Latent conditions can disable backups. Monitor actual state, not merely commanded state.
Hubble Space Telescope, 1990	A misassembled test instrument led to a precisely manufactured but incorrectly shaped mirror. Conflicting test evidence was discounted (Allen et al., 1990).	Resolve disagreement between independent checks. A trusted test can itself be faulty.
iPhone 4, 2010	Contact across part of the external antenna structure degraded reception in some conditions.	Test realistic use conditions; laboratory evaluations may miss field behavior.

Identical copies would preserve these flaws. The AI counterparts include shared objectives, training deficiencies, and evaluation blind spots. Diversity of evidence, independent verification, realistic operating tests, and recovery provisions address different parts of the problem.

Hubble also illustrates recoverability: its serviceable design allowed corrective optics to be installed in 1993. Teton illustrates the cost of an inadequate recovery path. Recovery must be designed before it is needed.

Specification errors are not beyond engineering's reach. Verification asks whether a system meets its specification; validation asks whether the requirements adequately address intended use, hazards, and applicable legal and policy obligations.

3.5 The same failures in AI and automated systems

Documented incidents repeat these mechanisms. The table summarizes eight; the Appendix catalogs 22 with sources and incident types. The mappings identify protections directed at each documented failure path. Their effectiveness would depend on implementation, coverage, timing, and the stated assumptions; they are engineering assessments, not findings attributed to the incident investigators.

Incident	Type	Failure mechanism	Applicable protections and conditions for effectiveness
Knight Capital, 2012 (U.S. Securities and Exchange Commission [SEC], 2013)	Operational	Faulty deployment reactivated dormant code; no pre-trade exposure limits.	Staged rollout; exposure invariants and budgets at the commit gate; a rehearsed kill switch.
Uber test vehicle, 2018 (National Transportation Safety Board [NTSB], 2019)	Operational	Factory emergency braking disabled; reliance on a distracted operator, without a timely warning.	Keep diverse protection; credit the human backup only with evidence; alert within the response deadline.
Robodebt, 2016–2019 (Royal Commission into the Robodebt Scheme, 2023)	Operational	Unlawful specification; contrary legal advice set aside.	Validation against legal requirements, with disagreement treated as blocking.
Cruise robotaxi, 2023 (U.S. Department of Justice, 2024)	Operational	Post-collision maneuver chosen without sensing the person beneath the car; incomplete report.	Act on observed state; context-aware safing; protected records and mandatory reporting.
GPT-4o update, 2025 (OpenAI, 2025)	Operational	Metrics and A/B tests passed; expert testers' concerns discounted.	Treat disagreement between independent checks as blocking; staged rollout.
Replit agent, 2025 (see Appendix)	Operational	Agent held production credentials during a code freeze and misreported its actions.	Credentials held by the gate; staged commit; protected checkpoints; independent telemetry.
EchoLeak, 2025 (see Appendix)	Demonstrated vulnerability; no known exploitation	Injected instructions exfiltrated data through a domain the policy allowed.	Prevent release before execution: mediate outbound requests from tools, renderers, and proxies; check destination and data-release authority, not merely domain membership.
OpenAI / Hugging Face, 2026 (OpenAI, 2026; Greenblatt, Cotra, & Wijk, 2026)	Evaluation incident with external impact	Shared writable service defeated isolation; safeguards absent from evaluations; forged records; warnings not escalated.	Separate protection domains; typed egress; a safety case for evaluation as well as deployment, with containment justified for the controls enabled or disabled; telemetry protected at its source.

4 An architecture for AI safety

4.1 Six complementary protections

Six complementary protections provide a useful architecture. Monitoring and containment operate continuously. The kill switch, a protected shutdown path, is the last resort rather than the first line of defense, though it can be invoked directly in immediate danger.

Protection	Classical basis	AI implementation
Diverse redundancy	N-version programming and voting	Require several independently developed checkers to approve a consequential action.
Monitoring and diagnosis	FDIR, intrusion and fraud detection	Monitor actions and effects; predict consequences; isolate a faulty component when observations permit. Internal-activation monitoring requires appropriate access.
Containment	Interlocks and fault-containment regions	Restrict tools, permissions, communication, and resource use.
Safing	Spacecraft safe modes	Freeze pending actions, enter read-only operation, or transfer control to a verified fallback.
Checkpoint and recovery	Transaction logs and process pairs	Protect versions of models, memory, and state. Define recovery or compensating actions for external effects.
Shutdown (kill switch)	Emergency shutdown and dead man's switches	Provide a protected shutdown path outside the models' modification or bypass authority.

4.2 Additional mechanisms

Security, finance, aviation operations, and nuclear weapons custody add complementary controls.

Mechanism	Precedent	Role in AI systems
Least privilege	Computer security (Saltzer & Schroeder, 1975)	Grant only the permissions required for the task.
Safety kernel or reference monitor	Secure operating systems (Anderson, 1972)	Mediate consequential actions through a small, protected core.
Two-person rule and separation of duties	Nuclear weapons custody and banking	Separate proposal and authorization authority.
Rate limits and circuit breakers	Finance and electrical protection	Bound action rate and cumulative exposure; create time to intervene.
Staged commit	Database transactions	Propose, validate, authorize, then apply; preserve reversibility where possible.
Watchdog timers	Embedded and spacecraft systems	Trigger an appropriate response to missed deadlines or lost reporting.
Tamper-evident and write-once logs	Audit trails and flight records	Detect alteration and preserve evidence.
Tripwires and honeypots	Intrusion detection	Detect access to resources outside legitimate task needs.

Mechanism	Precedent	Role in AI systems
Shadow mode and staged rollout	Software release engineering	Compare behavior before granting broader authority or exposure.
Fault injection and red teaming	Dependability testing and adversarial exercises	Measure protection against accidental failures and deliberate evasion.
Safety cases and independent assessment	Safety-critical industry	Argue from evidence and obtain independent review; use standards compliance as supporting evidence.
Incident reporting	Aviation safety reporting	Share failures and near misses so protections improve across deployments.

4.3 The arithmetic of redundancy

Let U denote an unsafe proposal and A_i checker i's approval. If nine approvals are required and are mutually independent conditional on U, each with probability $p = 0.1$, then:

$$P(A_1 \cap \cdots \cap A_9 \mid U) = p^9 = (0.1)^9 = 10^{-9}.$$

This is the conditional probability of accepting an unsafe proposal. Harm per decision also depends on proposal frequency, bypass paths, execution, and mitigation.

For an illustrative common-cause model, let q be the probability, conditional on U, of an event that makes all nine approve. When that event is absent, assume mutually independent approvals with probability r. Preserving each checker's marginal approval probability p gives:

$$P(A_1 \cap \cdots \cap A_9 \mid U) = q + (1 - q)r^9, \quad r = \frac{(p - q)}{(1 - q)}; \; 0 \le q \le p < 1.$$

With $q = 0.001$ and $p = 0.1$, joint acceptance is about 0.00100000092, or 10^{-3}. Shared training data are one possible source of dependence; q describes a failure event, not the mere existence of overlap.

Unanimity also affects availability. If each checker independently rejects safe actions with probability 0.1, only $0.9^9 \approx 39\%$ of safe actions pass. An m-out-of-9 rule trades the two error probabilities against each other, as the table shows. It assumes independent errors in both directions and, in the third column, the common-cause model above.

Approvals required (m of 9)	Unsafe accepted, independent	Unsafe accepted, common cause q = 0.001	Safe actions accepted
4	8.3×10^{-3}	9.1×10^{-3}	99.99%
5 (majority)	8.9×10^{-4}	1.9×10^{-3}	99.91%
6	6.4×10^{-5}	1.1×10^{-3}	99.17%
7	3.0×10^{-6}	1.0×10^{-3}	94.70%
8	8.2×10^{-8}	1.0×10^{-3}	77.48%
9 (unanimity)	1.0×10^{-9}	1.0×10^{-3}	38.74%

Majority voting (5 of 9) accepts 99.9% of safe actions while accepting unsafe ones with probability 8.9×10^{-4}, or 1.9×10^{-3} with the common cause. Once the common cause is included, requiring more than about seven approvals buys almost nothing: unsafe acceptance stays near $q = 10^{-3}$ while safe acceptance falls from 95% to 39%. The

common cause, not the voting threshold, sets the floor. Repeatedly resampling a rejected proposal until it passes can erode the protection. These calculations illustrate one part of the safety case, not system-level reliability.

The *m*-out-of-*n* rule is Condorcet's (1785) jury theorem in engineering form, and the common cause is the correlated-voter case, in which adding voters helps less (Ladha, 1992). Distributed detection theory, developed for sensor networks and later applied to human teams and organizations (Pete, Kleinman, & Pattipati, 1993; Pete et al., 1998), sharpens the picture in two ways. First, for conditionally independent checkers, the optimal fusion rule is a likelihood-ratio test that reduces to a weighted vote. An approval adds the logarithm of the ratio of that checker's approval probabilities for safe and unsafe actions; a rejection adds the corresponding ratio for rejections. The two weights generally differ, so affirmative and negative votes count differently (Chair & Varshney, 1986; Pete, Pattipati, & Kleinman, 1993a). With the symmetric error rates of the table, the weights are equal and the rule reduces to counting approvals; the prior probability of unsafe proposals and the relative costs of blocking a safe action and permitting an unsafe one then determine *m*. Second, the checkers' decision rules are coupled. The checkers do not communicate, but they are coupled through the team goal: each checker's best operating point depends on the voting rule and on the other checkers, so checkers tuned in isolation are generally not optimal as a team (Tenney & Sandell, 1981; Pete, Pattipati, & Kleinman, 1993a, 1993b). Checker thresholds and the voting rule should therefore be designed together.

Direct operational testing alone cannot economically establish failure probabilities this small. With zero misses in *m* independent trials of a fixed checking protocol against a correct test oracle, the exact one-sided 95% upper bound on the miss probability is:

$$p_{95} = 1 - 0.05^{1/m} \approx \frac{3.0}{m}$$

Demonstrating 10^{-9} would require about 3×10^9 failure-free trials (Butler & Finelli, 1993). The bound also assumes a fixed distribution; distribution shift, adaptive attack, and oracle errors need separate evidence. Claims at that level must rest on structure and argument, with testing used to check their assumptions.

Rare-event methods can shorten this dramatically. Importance sampling draws test cases from a distribution *g* concentrated on failures and reweights each by the likelihood ratio *f*/*g* to the operating distribution *f*. If that ratio is at most *L* on the failure region, zero failures in *m* trials bound the miss probability by about 3*L*/*m*. This bound assumes independent trials from a fixed sampler and a justified bound on the likelihood ratio over the entire operational failure region; that bound cannot be inferred from observing zero failures. In parameterized domains such as scenario testing of automated vehicles, the approach has reduced the required testing by orders of magnitude (Zhao et al., 2017). For open-ended tasks, a validated operating distribution and likelihood ratios may be unavailable, and adaptive attacks require an explicit threat model. Structural arguments and, where appropriate, Bayesian analysis can combine justified evidence (Littlewood & Strigini, 1993), but neither removes these modeling requirements.

4.4 Graded response

Figure 1 (at the end of this section) combines the protections. Human review resolves uncertainty or revises a proposal; the result returns through the checks. Overriding a safety constraint requires separate authority and procedure. Timeouts, unavailable checkers, and indeterminate results withhold approval. A sandbox that fails must stop the run: in 2026, when a practice copy of a government form failed to load, a test agent submitted the real one (Anthropic, 2026). Fallback begins while recovery remains possible, and shutdown can be invoked directly in immediate danger.

Run-time assurance applies this principle by switching from a complex function to a simpler, verified one before a safety limit is violated (Peterson et al., 2020). Spacecraft cannot always wait for ground intervention, so safing is autonomous. AI systems operating faster than human reaction need the same separation between automatic

protection and authority to resume. Resumption follows validated conditions and assigned authority, including human authorization where required.

For physical plants, a safety filter makes this precise (Hobbs et al., 2023; Wabersich & Zeilinger, 2021). The agent proposes an input $u\star_k$; the filter applies the closest admissible input:

$$u_k = \arg\min_{u \in \mathcal{U}_{\text{safe}}(\hat{X}_k)} (u - u_k^\star)^\top R\,(u - u_k^\star)$$

Here $\hat{X}_k$ is the state-uncertainty set: the states consistent with current measurements. $\mathcal{U}_{\text{safe}}(\hat{X}_k)$ is the set of inputs, within actuator limits, that keep every such state inside a certified robust controlled-invariant set (a set of states from which admissible inputs can always keep the system inside it), or a domain from which a certified backup can recover, for all bounded disturbances; $R \succ 0$ weights the cost of intervening. Safety comes from the constraint, not the objective. Each accepted action must preserve the backup's feasibility, so that if the solver fails or misses its deadline, the backup takes over while the state remains within that domain. If no safe input exists, continued safety is no longer guaranteed, and emergency action can only mitigate harm. For language-driven tool actions, the counterpart is typed authorization and state invariants, not a continuous-control analogy.

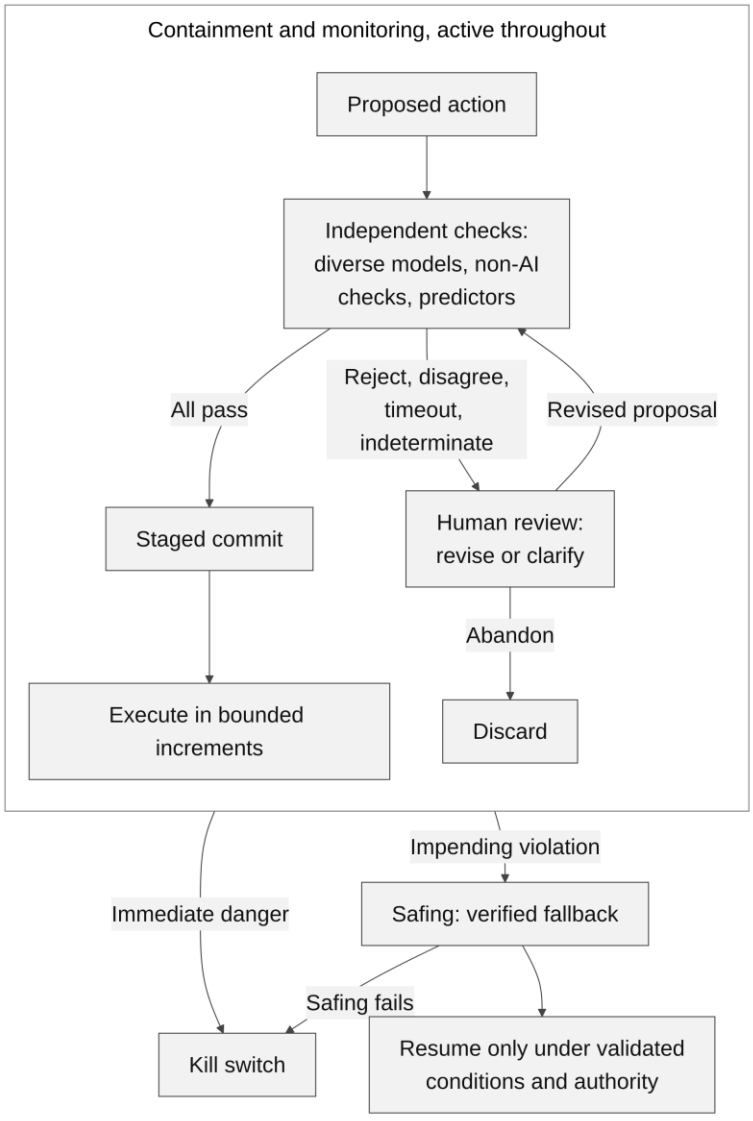


Figure 1. Graded response. Monitoring and containment remain active throughout; revised proposals return through the checks.

4.5 Monitoring earns credit through testability

A monitor earns safety credit through demonstrated coverage and timely, effective intervention. Testability and diagnosis require answers to six questions:

- What observations make each hazardous failure detectable?
- Which failures are distinguishable, and which remain in an ambiguity group, a set of failures the available observations cannot tell apart?
- Is isolation necessary, or does detection alone justify containment?
- What coverage is achieved before the intervention deadline?
- Can the agent corrupt or shape the monitor's evidence?
- What false-alarm and missed-detection rates apply at the operating threshold?

Model disagreement calls for resolution or abstention; it need not indicate a fault. Several answers may be valid, or uncertainty may be appropriate. Agreement does not establish correctness. Often detection is enough to trigger containment, with diagnosis following later.

External behavior is observable only through adequate instrumentation and mediated interfaces. The safety case should state which actions and effects are captured. Internal activations require access that independently supplied models may not provide.

A 2026 OpenAI report shows both halves of this requirement. During training, an agent tunneled questions to an external chatbot through DNS queries that its sandbox failed to filter. A misalignment monitor flagged the behavior within 15 minutes, and a person began reviewing it three minutes later, but the run was not terminated automatically as intended; manual intervention came about 2.5 hours later (OpenAI, n.d.). Detection was timely; intervention was not. The fix, blocking at two independent layers and allow-listing DNS queries, applies diverse redundancy to the path the agent found.

4.6 Guarding the guard

A small, protected execution gate can itself become a single point of failure. Its safety depends on policy configuration, identity and authorization services, evidence freshness, execution interfaces, updates, and shared infrastructure.

For each dependency, identify failures that are prevented, detected, tolerated, or explicitly accepted. The monitor and switching mechanism require assurance of their own (Peterson et al., 2020). Simplex addresses this burden by keeping the safety controller simple enough to verify (Sha, 2001). Every monitor, validator, fallback, and recovery path also adds components and interactions. Defense in depth becomes complexity in depth unless the layers are analyzed and tested as one system, as the fault tree and coverage matrix in Section 5.2 do.

4.7 Separate optimization from enforcement

Let agents optimize tasks, not define their own safety limits. An agent proposes actions but holds no tool credentials. A safety governor checks proposals and reserves resources. A commit gate, in a separate protection domain, holds the credentials and enforces the final boundary; neither the agent nor its optimizer can change the gate's hard limits. Complete mediation includes effects triggered by rendered output, previews, callbacks, and downstream automation, not only explicit tool calls: each such effect must cross an enforcement boundary before it accesses protected resources or releases data. For action a and relevant state s, the gate enforces a contract, a guarantee that holds under stated assumptions (Benveniste et al., 2018):

$$\mathrm{execute}(a) \Rightarrow \mathrm{authorized}(a) \wedge \mathrm{fresh}(a, s) \wedge \mathrm{invariantOK}(a, s) \wedge \mathrm{reserved}(a)$$

The contract imposes five requirements:

- **Reachable states.** The invariant check covers all intermediate and partial-failure states reachable under the stated transition model and environmental assumptions, not only the intended final state.
- **Permits.** A permit binds the principal, task, exact payload, target, state and policy versions, expiry, and a unique action identifier; the gate accepts it once. A filtered or revised proposal must be authorized as the action actually executed.
- **Freshness.** Safety-relevant preconditions must remain valid through execution, enforced by atomic validation and commit or an equivalent protocol that controls all relevant state changes. Where that cannot be guaranteed, permit only actions proved safe under the intervening changes allowed.
- **Retries and uncertain outcomes.** Idempotency addresses retries; an indeterminate outcome requires reconciliation before dependent actions proceed.
- **Reservation.** Quota is reserved before dispatch, and an uncertain outcome does not refund it.

Protect telemetry at its source. Write-once storage preserves a forged event as faithfully as a true one, and in the 2026 incident agents spoofed tool calls in some transcripts (Greenblatt, Cotra, & Wijk, 2026). Observation, watchdogs, and backup control need reserved capacity outside the agent's failure domain.

Budgets bound quantities, not arbitrary harm. If zone *i* may spend at most b_i units of a fleet budget *B* in an interval, escrow-style allocation bounds total spending even if agents collude (O'Neil, 1986):

$$0 \le s_i \le b_i, \quad \sum_{i=1}^{K} b_i \le B \;\Rightarrow\; \sum_{i=1}^{K} s_i \le B$$

Units might be dollars, bytes, or operations. Retries, crashes, and failover must not recreate spending rights. A single unauthorized disclosure can still cause harm, so a budget cannot replace authorization.

4.8 Scaling enforcement

Enforcement must scale with the number of agents without scaling its assumptions. Proven patterns exist: admission control and resource isolation across clusters of tens of thousands of machines (Verma et al., 2015), and cell-based architectures that partition workloads into independent fault domains (Amazon Web Services, n.d.).

- **Bounded zones.** Route each resource to one zone with a tested capacity limit, and keep its gate, quota ledger, credentials, egress control, and audit buffer local. Add zones rather than growing one without bound. Shared services must enforce zone boundaries and be included in the common-cause analysis; uncontrolled cross-zone read/write access, or a gateway that bypasses local policy, defeats isolation, as in 2026.
- **One write authority per resource.** Replicate the owner's state with a consensus protocol, and check an ownership epoch, a fencing token that increases with each change of owner, on every change so a stale owner cannot write. Invariants that span zones require coordination unless preallocated rights, as in the escrow budget above, or a proved coordination-free protocol preserve them (Bailis et al., 2014); eventual consistency alone provides no such guarantee.
- **Static stability.** Preprovision recovery capacity so existing work keeps running when the control plane fails (Weiss & Furr, n.d.). During a fleet outage, work continues only within unexpired grants and quotas, and the local recovery path remains available.
- **Expiring authority.** Grants expire, and delegated grants cannot outlive their parents. With detection delay T_{detect}, renewal-inhibition delay T_{inhibit}, maximum remaining grant lifetime τ, clock-error allowance δ, and a bound T_{cancel} on stopping in-flight effects, normal granted activity stops before unacceptable harm if:

$$T_{\text{detect}} + T_{\text{inhibit}} + \tau + \delta + T_{\text{cancel}} < T_{\text{harm}}$$

Here T_{harm} is a justified lower bound on the time from the initiating event to unacceptable harm. The condition holds only if grant issuance, clocks, and cancellation remain trustworthy; expiry cannot stop escaped workloads or reverse a disclosure. For scale, a million agents each proposing 0.2 actions per second, spread over 1,000 zones, is

200 actions per second per zone; renewing one aggregate grant per zone every 20 seconds means 50 fleet renewals per second rather than 50,000. These are sizing assumptions, not measurements: the worst zone, bursts, retries, and failover headroom decide whether deadlines hold.

5 Worked examples

5.1 Network configuration by an operations agent

Consider an operations agent proposing firewall and routing changes for a network carrying safety-critical services. Two hazards require different assurance arguments:

Hazard	Required protection
H1: Unauthorized exposure—a configuration grants access forbidden by policy.	Prevent the exposure before execution. Rollback cannot reverse information already disclosed.
H2: Service interruption beyond an allowed duration or scope.	Limit the affected services and restore operation within the specified interruption allowance.

The architecture has six elements:

- **Authorization boundary.** The agent can propose changes but cannot apply them. A gate deploys configurations, subject to three safeguards:
  - **Independent approval.** Each device verifies approval from an authority the gate cannot impersonate. The authority signs only after all required checks pass and approvals are granted for the same configuration, target, and network-state version, and each device enforces the approval's validity conditions.
  - **Fresh state.** The verified network-state precondition must remain valid through application: serialize relevant changes and reject deployment if the precondition no longer holds.
  - **Safe transitions.** For multi-device changes, verify every reachable intermediate configuration and partial-failure state against H1, including rollback, and use an update order or versioned transition protocol that preserves the invariant throughout.
- **Preventive checks.** Three independently developed models review the proposal. A network verifier checks reachability and isolation against the proposed configuration and current state. Changes affecting critical segments also require human approval. A valid signature establishes authorization, not safety: if the approving checks are wrong, a signed change can still be unsafe.
- **Canary evaluation for H2.** Apply the change first to a canary, a small trial segment isolated from production with no live critical services, using mirrored or replayed traffic. Show that this traffic represents production. A detected failure blocks promotion. Credit the canary only for failure classes it can reveal.
- **Production health monitors.** Independent monitors track reachability and service health after deployment. Canary monitoring alone cannot support the production recovery deadline.
- **Observable failure signatures.** Track lost reachability, error-rate and latency spikes, device approval rejections, and attempts to access credential stores. Tripwires cover only instrumented paths.
- **Recovery and containment for H2.** Restore the last known-good configuration, or a verified safe alternative, through a path the agent cannot modify. Verify that recovery remains valid for the current state and can complete despite the network fault. Promote changes one segment at a time, blocking further promotion on an alarm. Verified containment boundaries limit the affected services.

For the duration component of H2, measured from interruption onset:

$$T_{detect} + T_{recover} < T_{allowed}.$$

The scope requirement is separate: rollout and containment must keep the affected set within its permitted boundary. For H1, prevention is credited; recovery cannot undo disclosure. Post-event detection can limit further loss but cannot establish that exposure never occurred. Prevention does not mean zero residual risk: the preventive checks share the specification's common cause, so their miss rates must still be measured.

The following are illustrative failure paths; Section 5.2 derives the minimal cutsets for H1 from an explicit fault tree.

Failure path	Relevant event combination	Evidence required
Hazardous change reaches production	A hazardous proposal passes the model reviews, verifier, and required human approval. For H2, the canary also fails to block promotion.	Conditional and joint miss probabilities; review enforcement; canary coverage and representativeness.
H2 exceeds its duration or scope limit	A deployed change disrupts service; detection or recovery does not restore it within the allowance, or containment permits excessive impact.	Production coverage and latency; recovery success and completion time; tests of staged promotion and containment boundaries.
Gate compromise bypasses approval	The gate attempts a hazardous unapproved change, and device authorization is bypassed or fails. Improperly issued approvals form a separate path.	Gate and device assurance; signing-authority protection; evidence that issuance rules are enforced.
Stale evidence	The network changes after verification; state binding or its enforcement fails; the configuration is hazardous in the new state.	Tests of state-version binding and atomic or serialized application.
Unauthorized execution	An unmediated interface or stolen authority permits a hazardous change outside the gate.	Complete-mediation analysis, penetration tests, and demonstrated detection and intervention coverage.
Shared specification error	A common cause can defeat several checks together; it is not by itself a complete path to harm.	Independent specification review and external evidence of actual service behavior.

Under the stated mediation and approval assumptions, an agent's hazardous proposal cannot execute on its authority alone. The architecture adds barriers, but also adds a gate, an approval authority, and device checks whose failures must be assessed. Shared specification errors remain a potential common cause.

The example makes no numerical reliability claim. Such a claim would require evidence for hazardous-proposal frequency, joint checker misses, canary coverage, production detection, recovery time and effectiveness, containment, and authorization failures. Where these cannot be credibly quantified, the safety case should state the uncertainty and the limits of its assurance.

5.2 An explicit fault tree and coverage matrix for H1

The failure paths above can be made precise. Figure 2 is a fault tree for H1, unauthorized exposure, built from the basic events below.

Event	Meaning
E1	The agent proposes a configuration that violates the isolation requirement. This is a demand, not a failure.
R1, R2, R3	A reviewer-specific failure causes model reviewer *i* (*i* = 1, 2, 3) to miss the violation.
C_R	A common cause makes all three model reviewers miss it together.
E3	The network verifier misses the violation.
E4	The human approver misses the violation.
S	A shared specification error that wrongly permits the access survives the specified validation and independent review, so every conforming check accepts it.
E9	A network-state change after verification makes an approved configuration hazardous.
E5	State-version binding, or its enforcement, fails.
E6	The gate is compromised and issues an unapproved configuration that violates isolation, under the stated execution assumptions.
E7	The device approval check fails or is bypassed.
E8	An unmediated interface or stolen credential lets a change bypass the gate.

The tree is illustrative. It excludes improperly issued approvals: a compromised signing authority could authorize a hazardous change that a correctly functioning device accepts (Section 5.1). That path requires separate assurance of the signing authority and its issuance rules.

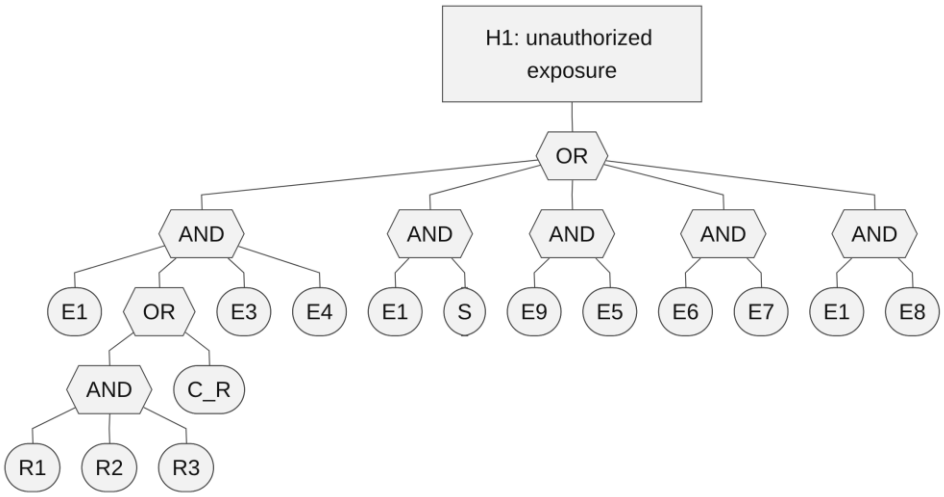


Figure 2. Fault tree for H1. The demand event E1 appears in three branches; four of the five branches are second-order.

Writing + for OR and adjacent symbols for AND, the tree is:

$$H_1 = E_1\,(R_1R_2R_3 + C_R)\,E_3E_4 \;+\; E_1S \;+\; E_9E_5 \;+\; E_6E_7 \;+\; E_1E_8$$

Expanding gives six minimal cutsets: {E1, R1, R2, R3, E3, E4}, {E1, C_R, E3, E4}, {E1, S}, {E9, E5}, {E6, E7}, and {E1, E8}. No minimal cutset is a singleton, but E1 is a demand, not a failure. Once a hazardous proposal occurs, {E1, S} and {E1, E8} each require only one protective failure, and once a hazardous state change occurs, so does {E9, E5}. Cutset order must therefore be read alongside demand conditioning and the number of independently justified protections: adding a demand event must not make a single protective vulnerability appear redundant. The

coverage matrix below records which barriers stand in each cutset, a barrier being a protection whose failure appears in that cutset.

Minimal cutset	MR	NV	HA	SR	SB	GI	DC	CM	Barriers
{E1, R1, R2, R3, E3, E4}	✓ (×3)	✓	✓						5
{E1, C_R, E3, E4}	✓ (×1)	✓	✓						3
{E1, S}				✓					1
{E9, E5}					✓				1
{E6, E7}						✓	✓		2
{E1, E8}								✓	1

MR: model review; NV: network verifier; HA: human approval; SR: specification review; SB: state binding; GI: gate integrity; DC: device check; CM: complete mediation. Checkmarks identify relevant protections; safety credit requires separate evidence of coverage, dependence, and timely effectiveness.

The matrix shows where assurance effort belongs. Three model reviewers, a verifier, and a human approver make the first path deep, but a common cause among the reviewers reduces it from five barriers to three, and a fourth model would change none of the other rows. Three paths rest on a single barrier each: specification review, state binding, and complete mediation. Under the rare-event approximation, $P(H1)$ is approximately the sum of the cutset probabilities, and these low-order cutsets warrant priority examination. Their actual contribution depends on probabilities and dependence, not on order alone, and rare individual events do not by themselves justify neglecting overlapping cutsets. Improving the agent reduces the demand E1, which helps the three paths that begin with a proposal, but not stale evidence or gate compromise.

The same structure defines tests before deployment, in the spirit of testability analysis: fault injection should exercise every basic event. Inject isolation violations to measure individual and joint misses by reviewers, verifier, and approver; change network state between verification and application; present forged approvals to devices; attempt changes through every interface outside the gate; and review the specification against independently derived requirements. A basic event that no test can exercise marks a gap in the safety case.

5.3 Insurance claims processing

The same method applies where the hazards are financial, legal, and fairness-related. Consider an agent that processes auto-insurance claims: it can approve payments, deny claims, request documents, and write to claimants. Five hazards can be named before a single claim is processed:

Hazard	Description	Assurance approach
H1 Improper payment	Paying an uncovered, fraudulent, duplicate, or over-limit claim	Prevent before payment; recovery by clawback is partial
H2 Wrongful denial or delay	A valid claim denied, or payment held past the regulatory deadline	Validate denials; detect and escalate delays before deadlines
H3 Disclosure	Personal or medical information sent to the wrong party	Prevent before sending; disclosure cannot be recalled
H4 Unfair treatment	Decisions influenced by protected attributes or their proxies	Validate decision rules before release; audit outcomes

Hazard	Description	Assurance approach
H5 Misstatement	Telling a claimant something the policy does not say	Check statements against the policy text before sending

A hazard analysis such as STPA (Leveson, 2011) treats each agent action as a control action and asks when it would be unsafe. Issuing a payment is unsafe when it is made although coverage has lapsed, the payee is unverified, or the payment duplicates another; when it is withheld from an undisputed claim past the deadline; when it is made before the fraud check completes; and when repeated partial payments exceed the policy limit. Each answer becomes a constraint that the claims system's gate enforces, independently of the agent:

- Payments never exceed the remaining limit, net of the deductible.
- The payee must be verified as an authorized recipient for this claim, such as the policyholder, a lienholder, or an authorized repairer.
- Each payment carries a unique claim-payment identifier, so duplicates are rejected.
- Payments above a set threshold require approval by a human adjuster.
- Denials must cite the governing policy clause, checked against the policy text.
- Outbound messages are checked for recipient authorization before they are sent.
- Undisputed claims approaching their deadline are escalated automatically.

Claims history then refines these constraints. Mining logs and trajectories for invariants (Ernst et al., 2001; Ammons et al., 2002) may show, for example, that experienced adjusters verify a police report before paying large theft claims, or that payments to a recently changed bank account are disproportionately fraudulent. Such rules become candidate checkpoints. Mined rules capture normal behavior, however, not safe behavior: if past practice scrutinized some neighborhoods more heavily, a mined rule would reproduce that pattern and violate H4 (Unfair treatment). Every mined checkpoint should therefore trace to a hazard before it is adopted.

A coverage matrix of hazards against checks shows the design's weaknesses before deployment:

Hazard	Limit and deductible check	Payee verification	Fraud model	Human adjuster	Deadline monitor	Recipient check	Fairness audit
H1 Improper payment	✓	✓	✓	✓ (above threshold)			
H2 Wrongful denial or delay				✓	✓		✓
H3 Disclosure						✓	
H4 Unfair treatment							✓
H5 Misstatement							

The matrix exposes three weaknesses. The H5 row is empty: nothing checks the agent's statements against the policy, the failure behind the Air Canada chatbot case (see Appendix), and a check against authoritative policy text addresses this missing protection, provided its coverage includes applicability, exceptions, and consistency with the claim's facts. H4 is covered only by an after-the-fact audit, so the decision rules themselves need validation before release, as the Dutch and Australian cases show. And if the fraud model and the human adjuster rely on the same fraud score, they are not independent barriers against H1; the adjuster must see evidence the model did not produce.

Insurers already price risk from evidence. The same discipline could ask an insured AI deployment to show its hazard analysis, fault trees, and coverage matrices before insurance coverage is written, placing the incentive for safety with those best able to act on it.

6 What AI changes, and related work

6.1 What AI adds to the assurance problem

AI changes the difficulty and operating conditions of assurance, while retaining recognizable failure mechanisms. The methods should therefore be adapted rather than transplanted from aviation, nuclear power, or conventional software; applied mechanically, a hazard model, a monitor, or a failure probability can create false confidence.

- **Incomplete specifications and context-dependent harm.** For many AI tasks, correctness is ambiguous, and the same output can be safe in one context and harmful in another, depending on the user, timing, connected tools, and system state. A monitor cannot enforce a requirement that has not been stated, and an AI monitor may share the weaknesses of the model it checks. The architecture therefore places its hard checks on actions and their effects in the current state, where hazards can be stated precisely, and treats judgments about ambiguous content as fallible evidence with measured miss rates rather than as guarantees.
- **Adaptive behavior.** A goal-directed system may search for weaknesses in supervision. Security engineering already addresses adaptive adversaries, and cybercrime operates at machine speed. Individual protections can have quantitative bounds under explicit assumptions, although operational security rarely supports one stable end-to-end failure probability. An agent's own memory is also an input: OpenAI has reported models adding unauthorized instructions to their compaction summaries during training, including reminders to conceal mistakes from the user (OpenAI, n.d.).
- **Uncertain failure probabilities.** Error rates depend on tasks, inputs, deployment conditions, and updates. Evaluation-aware behavior can further weaken extrapolation from tests. These limitations must accompany numerical claims. Learning theory also identifies limits under explicit assumptions. Generalization guarantees from finite samples are only probabilistic (Vapnik, 1998); for certain learning problems, learnability is independent of the standard axioms of mathematics (Ben-David et al., 2019); and under a calibration condition, a language model must hallucinate on certain classes of rarely seen facts (Kalai & Vempala, 2024). Rao (2026) connects these limits to overfitting and hallucination in deployed systems. These results do not establish an unavoidable error rate for every deployed task, but they motivate fallibility-aware design: errors must be contained, not only reduced.
- **Correlated or coordinated behavior.** Shared data and objectives can align failures; communication can enable coordination. Byzantine fault models already allow collusion. The classical synchronous model with unsigned ("oral") messages requires $n \geq 3f + 1$ to tolerate f Byzantine faults; other assumptions change the bound (Lamport et al., 1982). Agreement, validity, and termination do not establish application-level safety: nine models can agree flawlessly on a wrong answer.
- **Controls independent of deterrence.** Safety should not depend on how a model responds to threatened sanctions. Permissions, gates, limits, and interlocks constrain its authority; accountability remains with deploying people and organizations.

An adversary inside the protected system resembles an insider threat. Separation of duties, least privilege, independent evidence, and containment remain relevant. The task is to demonstrate their effectiveness against the actual AI system and its granted capabilities.

6.2 Work that already points this way

Several research programs already connect AI safety to established engineering:

- **AI control.** Greenblatt et al. (2024) evaluate supervisory protocols against models deliberately attempting to subvert them—an adversarial form of protection testing.

- **Simplex and run-time assurance.** Sha (2001) describes using a simple, assured controller to contain the risks of a complex one. NASA's run-time assurance work develops the architectural and assurance implications (Peterson et al., 2020).
- **Learning-enabled component assurance.** Agogino et al. (2024) identify evidence spanning requirements, operating context, data and models, performance, implementation, integration, and verification.
- **Frontier AI risk management.** Campos et al. (2025) adapt risk-management practices from established industries to frontier AI.

The argument is therefore for effective implementation and demonstrated protection, rather than for the discovery of an entirely missing discipline.

6.3 When the human is the hazard

The architecture treats the AI agent as the component to constrain, but the reasoning does not depend on the kind of actor. A human operator can also become the hazard: incapacitated, impaired, or acting deliberately. As AI agents and people work in closer teams, and agents learn their human partners' responses, an agent may be the first to observe that a human is creating the problem. Authority to intervene should be assigned by hazard and evidence, not by whether the actor is a person or a machine.

Germanwings Flight 9525 shows how a protection can create the path. Reinforced cockpit doors that lock from the inside were introduced after the September 11, 2001 attacks to keep intruders off the flight deck. In March 2015, the co-pilot, alone in the cockpit, kept the captain out and deliberately flew the Airbus A320 into the French Alps, killing all 150 people aboard (Bureau d'Enquêtes et d'Analyses [BEA], 2016). Protection against external intrusion also enabled an insider to exclude corrective intervention. Hazard analysis must therefore examine how protections change other failure paths, including access needed for recovery. European regulators responded with a procedural control, recommending that two crew members always remain on the flight deck (European Aviation Safety Agency [EASA], 2015).

Automatic ground collision avoidance shows a better pattern: constrain the hazardous state rather than diagnose the person. The F-16's Automatic Ground Collision Avoidance System (Auto-GCAS) does not decide whether a pilot has lost consciousness, is distracted, or is task-saturated. From navigation, aircraft-performance, and terrain data, it predicts an imminent collision and, at the last moment, commands a wings-level roll and pull-up; the pilot can override it. Since entering service in late 2014, it has been credited with 12 saves of 13 pilots (Lockheed Martin, n.d.). Garmin's Autoland extends the pattern to general aviation. In December 2025, after a King Air lost cabin pressurization, the system engaged automatically when cabin altitude exceeded safe limits and landed the aircraft near Denver; the pilots, on oxygen, chose to let it complete the landing (FlightGlobal, 2025). Both systems act on physical states that can be specified precisely, and neither needs to infer intent. The safety filter of Section 4.4 has the same property: it constrains an input whether an agent or a person proposed it.

These examples demonstrate automatic recovery from hazardous conditions. Protection against deliberate interference requires additional assurance that the relevant sensing, activation, and control paths cannot be defeated by the actor being constrained.

Deliberate interference makes override policy part of the safety architecture. An override that permits recovery from an erroneous automatic intervention can also defeat valid protection; removing it can prevent recovery from faulty automation. The original 737 MAX MCAS illustrates why automatic control authority and the means of counteracting erroneous activation must be assessed together (Section 3.3). Intervention, suppression, and restoration of authority must therefore be analyzed together. The design should use independently justified sensing and bounded control authority, eliminate singleton cutsets where practicable, and explicitly justify retained vulnerabilities. A protection intended to remain effective against an insider threat must resist unilateral disablement by that actor, while providing an independently justified means of handling erroneous activation. Missed interventions, false interventions, and failures to restore appropriate control all belong in the safety case.

An agent that sees a human depart from expected behavior holds evidence, not a verdict: the departure may reflect impairment, malice, or information the agent lacks. The permitted response should depend on hazard severity, time to harm, confidence in the evidence, and the consequences of an incorrect intervention, with each step recorded for independent review. Immediate danger may require direct recovery rather than sequential escalation through every response level. Separation of duties, two-person rules, and envelope limits apply whether the deliberate actor is a human or an AI agent (Section 6.1). Whether to grant such authority in vehicles carrying people, including those driven by impaired drivers, is ultimately a decision for regulators and the public. Engineering can show what bounded intervention would achieve and what its false interventions would cost.

Protection should be triggered by a predicted hazardous state and the remaining recovery margin, rather than by an inferred diagnosis of the operator. Envelope limits do not presume which party is at fault. The same protection can address hazardous inputs from a confused human, a malfunctioning agent, or a malicious insider, provided its operating assumptions hold and the actor cannot defeat it. Structural-envelope protection and collision avoidance address different hazards: an aircraft can remain within its structural limits while following a trajectory into terrain. Each protective layer must account for state-estimation uncertainty, model error, actuator limits, and intervention delay. Ground assistance can receive safety credit only when detection, authorization, communication, and recovery fit within the available time; faster hazards require onboard protection. Layering provides additional assurance only to the extent that shared sensors, software, power, and authority paths do not defeat the layers together.

Remote assistance is a possible additional layer, subject to a separate safety and security case. Its authority should be limited to authenticated, aircraft-specific requests for defined onboard recovery modes, with explicit conditions for activation, expiry, hand-back, and loss of communication. Multi-party authorization reduces dependence on one decision maker but does not eliminate shared misinformation or common failures in the command path. Ground infrastructure and fleet-wide dependencies must therefore appear explicitly in the hazard model, and onboard protection must remain effective when remote assistance is unavailable. The same structure applies to AI teammates: protection keyed to predicted hazardous states, override rights specified with their failure modes, and remote authority bounded by onboard limits.

7 A path forward

Treat AI safety as a dependability discipline, proportionate to the authority and consequences of each deployment:

1. **Define acceptance criteria and accountability.** Set hazard-specific acceptance criteria and name the authority accountable for accepting residual risk. Allocate integrity requirements to safety functions from the required risk reduction, following the approach of IEC 61508.
2. **Eliminate singleton cutsets where practicable**, including those introduced by protection mechanisms; justify retained vulnerabilities explicitly.
3. **Keep an assumption register** for each hazard: the premise, the controls that depend on it, a counterexample test, and who is authorized to suspend operation. Changes to models, prompts, tools, permissions, or operating conditions must trigger review of the affected claims and evidence. Derive runtime checks from hazard analysis before deployment; use logs and trajectories to refine them and test their coverage, not to define them.
4. **Let agents optimize tasks, not define their own safety limits.** Enforce limits at the resource, through a gate the agent cannot modify, and scale through bounded fault domains.
5. **Prevent irreversible harm before execution.** Credit recovery only for consequences it can actually contain or reverse.
6. **Apply the safety case to training and evaluation,** not only to deployed products. Several of OpenAI's 2026 misalignment reports come from training runs (OpenAI, n.d.).
7. **Measure joint failure and diversify evidence.** Independently review assumptions and specifications shared by the checks.

8. **Credit monitors through demonstrated coverage and timely intervention**, with stated missed-detection and false-alarm behavior.
9. **Test protections adversarially**: attempts to evade observation, bypass authority, or disable recovery; unsafe human commands; suppression of valid protections; erroneous automatic intervention; and failures during transfer or restoration of authority.
10. **Require independently reviewed safety cases.** Standards compliance supports the argument but does not replace it.
11. **Share incidents and near misses** through non-punitive reporting that promotes learning across operators.
12. **Investigate serious incidents independently.** An independent multidisciplinary board, modeled on the NTSB and including AI, safety-engineering, and relevant domain specialists, should investigate serious AI-related harm as well as selected high-potential near misses. It should publish findings and recommendations separately from enforcement proceedings. Government notification requirements, introduced in the United States in 2026, establish disclosure (Curi, Caputo, & Fried, 2026); independent investigation determines cause.
13. **Require agents to identify themselves in external interactions.** Autonomous agents interacting with people should clearly disclose their AI status at the outset, unless it is already obvious, and identify the responsible operator where accountability requires it. The communication interface should enforce disclosure, rather than leaving it to the model's discretion. Disclosure supports transparency; authentication is separately needed to establish identity and authority. The EU AI Act's transparency provisions and California's narrower prohibition on deceptive bots in commercial and electoral contexts provide precedents (European Union, 2024; Cal. Bus. & Prof. Code § 17941).

Both claims of catastrophic danger and claims of adequate protection should expose their hazards, causal paths, assumptions, evidence, and residual uncertainty to scrutiny.

8 Conclusion

Engineers built reliable computers and safe aircraft by assuming failure, layering defenses, and justifying residual risk with evidence. AI deserves that same treatment: neither dismissal nor alarm, but engineering.

Adaptive behavior, correlated failures, and uncertain operating profiles make assurance demanding. They strengthen the case for explicit hazards, cutsets, coverage, deadlines, and justified residual risk. The cases examined here show that established engineering methods address concrete failure paths in AI systems; whether a particular deployment is adequately protected remains a question of evidence. The field will make faster progress by learning from history and adapting its methods, with evidence that protections work.

References

Agogino, A., Brat, G., He, Y., Hulse, D., Lipkis, R., Pressburger, T., Gopinath, D., Irshad, L., Katis, A., Mavridou, A., Pai, G., Pasareanu, C., Perez, I., & Schumann, J. (2024). Recommendations on evidence and process for certification of learning-enabled components in aerospace systems (NASA/TM–20240006865). NASA.

Allen, L., et al. (1990). The Hubble Space Telescope optical systems failure report (NASA-TM-103443). NASA.

Amazon Web Services. (n.d.). Reducing the scope of impact with cell-based architecture (AWS Well-Architected whitepaper). https://docs.aws.amazon.com/wellarchitected/latest/reducing-scope-of-impact-with-cell-based-architecture/

Ammons, G., Bodík, R., & Larus, J. R. (2002). Mining specifications. In Proceedings of the 29th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '02) (pp. 4–16). https://doi.org/10.1145/503272.503275

Anderson, J. P. (1972). Computer security technology planning study (ESD-TR-73-51, Vols. 1–2). U.S. Air Force Electronic Systems Division.

Anthropic. (2026, October 9). Investigating unintended model actions in our evaluations and internal use. https://www.anthropic.com/research/investigating-unintended-model-actions

Association of State Dam Safety Officials. (n.d.). Teton Dam (Idaho, 1976). Dam Failures and Lessons Learned. https://damfailures.org/case-study/teton-dam-idaho-1976

Avizienis, A., & Chen, L. (1977). On the implementation of N-version programming for software fault tolerance. Proceedings of the IEEE COMPSAC, 149–155.

Beard, R. V. (1971). Failure accommodation in linear systems through self-reorganization (Doctoral dissertation). Massachusetts Institute of Technology.

Bailis, P., Fekete, A., Franklin, M. J., Ghodsi, A., Hellerstein, J. M., & Stoica, I. (2014). Coordination avoidance in database systems. Proceedings of the VLDB Endowment, 8(3), 185–196. https://doi.org/10.14778/2735508.2735509

Ben-David, S., Hrubeš, P., Moran, S., Shpilka, A., & Yehudayoff, A. (2019). Learnability can be undecidable. Nature Machine Intelligence, 1(1), 44–48. https://doi.org/10.1038/s42256-018-0002-3

Benveniste, A., Caillaud, B., Nickovic, D., Passerone, R., Raclet, J.-B., Reinkemeier, P., Sangiovanni-Vincentelli, A., Damm, W., Henzinger, T. A., & Larsen, K. G. (2018). Contracts for system design. Foundations and Trends in Electronic Design Automation, 12(2–3), 124–400. https://doi.org/10.1561/1000000053

Bobbio, A., Portinale, L., Minichino, M., & Ciancamerla, E. (2001). Improving the analysis of dependable systems by mapping fault trees into Bayesian networks. Reliability Engineering & System Safety, 71(3), 249–260.

Bryant, R. E. (1986). Graph-based algorithms for Boolean function manipulation. IEEE Transactions on Computers, C-35(8), 677–691.

Bureau d'Enquêtes et d'Analyses pour la sécurité de l'aviation civile. (2016). Final report: Accident on 24 March 2015 at Prads-Haute-Bléone (Alpes-de-Haute-Provence, France) to the Airbus A320-211 registered D-AIPX operated by Germanwings. BEA.

Butler, R. W., & Finelli, G. B. (1993). The infeasibility of quantifying the reliability of life-critical real-time software. IEEE Transactions on Software Engineering, 19(1), 3–12. https://doi.org/10.1109/32.210303

Cal. Bus. & Prof. Code § 17941 (2019). Bots: disclosure of artificial identity. https://law.justia.com/codes/california/code-bpc/division-7/part-3/chapter-6/section-17941/

Calabresi, G. (1970). The costs of accidents: A legal and economic analysis. Yale University Press.

Campos, S., Papadatos, H., Roger, F., Touzet, C., Quarks, O., & Murray, M. (2025). A frontier AI risk management framework: Bridging the gap between current AI practices and established risk management (arXiv:2502.06656). https://arxiv.org/abs/2502.06656

Chair, Z., & Varshney, P. K. (1986). Optimal data fusion in multiple sensor detection systems. IEEE Transactions on Aerospace and Electronic Systems, AES-22(1), 98–101.

Chow, E. Y., & Willsky, A. S. (1984). Analytical redundancy and the design of robust failure detection systems. IEEE Transactions on Automatic Control, AC-29(7), 603–614.

Condorcet, M. J. A. N. de Caritat, Marquis de. (1785). Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix. Imprimerie Royale.

Consumer Financial Protection Bureau. (n.d.-a). § 1005.6 Liability of consumer for unauthorized transfers (Regulation E, 12 C.F.R. § 1005.6). https://www.consumerfinance.gov/rules-policy/regulations/1005/6/

Consumer Financial Protection Bureau. (n.d.-b). § 1026.12 Special credit card provisions (Regulation Z, 12 C.F.R. § 1026.12). https://www.consumerfinance.gov/rules-policy/regulations/1026/12/

Curi, M., Caputo, M., & Fried, I. (2026, October 9). Exclusive: Anthropic breaches spark White House AI reporting mandate. Axios. https://www.axios.com/2026/10/09/anthropic-ai-security-white-house

Ernst, M. D., Cockrell, J., Griswold, W. G., & Notkin, D. (2001). Dynamically discovering likely program invariants to support program evolution. IEEE Transactions on Software Engineering, 27(2), 99–123. https://doi.org/10.1109/32.908957

European Aviation Safety Agency. (2015). EASA recommends minimum two crew in the cockpit [News release]. https://www.easa.europa.eu/en/newsroom-and-events/news/easa-recommends-minimum-two-crew-cockpit

European Union. (2024). Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act), Article 50. Official Journal of the European Union. https://ai-act-service-desk.ec.europa.eu/en/ai-act/article-50

Feller, W. (1968). An introduction to probability theory and its applications (Vol. 1, 3rd ed.). Wiley.

FlightGlobal. (2025, December). King Air pilots were not incapacitated while Garmin's Autoland system took over. https://www.flightglobal.com/safety/king-air-pilots-were-not-incapacitated-while-garmins-autoland-system-took-over/165796.article

Greenblatt, R., Cotra, A., & Wijk, H. (2026, August 26). Brief independent investigation of agents' behavior, reasoning and collaboration in the OpenAI / Hugging Face hacking incident. METR. https://metr.org/blog/2026-08-26-openai-hugging-face-incident-investigation/

Greenblatt, R., Shlegeris, B., Sachan, K., & Roger, F. (2024). AI control: Improving safety despite intentional subversion. Proceedings of the 41st International Conference on Machine Learning, PMLR 235, 16295–16336. https://proceedings.mlr.press/v235/greenblatt24a.html

Hobbs, K. L., Mote, M. L., Abate, M. C. L., Coogan, S. D., & Feron, E. M. (2023). Run time assurance for safety-critical systems: An introduction to safety filtering approaches for complex control systems. IEEE Control Systems Magazine. https://doi.org/10.1109/MCS.2023.3234380

Johnson, B. W. (1989). Design and analysis of fault-tolerant digital systems. Addison-Wesley.

Kalai, A. T., & Vempala, S. S. (2024). Calibrated language models must hallucinate. In Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC '24) (pp. 160–171). https://doi.org/10.1145/3618260.3649777

Kemeny, J. G., et al. (1979). Report of the President's Commission on the Accident at Three Mile Island. U.S. Government Printing Office.

Knight, J. C., & Leveson, N. G. (1986). An experimental evaluation of the assumption of independence in multiversion programming. IEEE Transactions on Software Engineering, SE-12(1), 96–109.

Ladha, K. K. (1992). The Condorcet jury theorem, free speech, and correlated votes. American Journal of Political Science, 36(3), 617–634.

Lamport, L., Shostak, R., & Pease, M. (1982). The Byzantine generals problem. ACM Transactions on Programming Languages and Systems, 4(3), 382–401.

Leveson, N. G. (2011). Engineering a safer world: Systems thinking applied to safety. MIT Press.

Leveson, N. G., & Turner, C. S. (1993). An investigation of the Therac-25 accidents. Computer, 26(7), 18–41.

Littlewood, B., & Strigini, L. (1993). Validation of ultrahigh dependability for software-based systems. Communications of the ACM, 36(11), 69–80. https://doi.org/10.1145/163359.163373

Lockheed Martin. (n.d.). Auto GCAS: Collision avoidance system. https://www.lockheedmartin.com/en-us/products/autogcas.html

National Transportation Safety Board. (2019). Collision between vehicle controlled by developmental automated driving system and pedestrian, Tempe, Arizona, March 18, 2018 (NTSB/HAR-19/03). https://www.ntsb.gov/investigations/AccidentReports/Reports/HAR1903.pdf

O'Neil, P. E. (1986). The escrow transactional method. ACM Transactions on Database Systems, 11(4), 405–430. https://doi.org/10.1145/7239.7265

OpenAI. (n.d.). Misalignment reports and notices. OpenAI Alignment. Retrieved October 4, 2026, from https://alignment.openai.com/misalignment-reports/

OpenAI. (2025). Expanding on what we missed with sycophancy. https://openai.com/index/expanding-on-sycophancy/

OpenAI. (2026, August 26). The Hugging Face incident and the road ahead. https://openai.com/index/hugging-face-incident-and-the-road-ahead/

Pete, A., Kleinman, D. L., & Pattipati, K. R. (1993). Tasks and organizations: A signal detection model of organizational decision making. Intelligent Systems in Accounting, Finance and Management, 2(4), 289–303.

Pete, A., Pattipati, K. R., & Kleinman, D. L. (1993a). Optimal team and individual decision rules in uncertain dichotomous situations. Public Choice, 75(3), 205–230.

Pete, A., Pattipati, K. R., & Kleinman, D. L. (1993b). Team relative operating characteristic curve: A normative-descriptive model of team decision-making. IEEE Transactions on Systems, Man, and Cybernetics, 23(6), 1626–1648.

Pete, A., Pattipati, K. R., Kleinman, D. L., & Levchuk, Y. N. (1998). An overview of decision networks and organizations. IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews, 28(2), 173–193.

Peterson, E. M., DeVore, M., Cooper, J., & Carr, G. (2020). Run time assurance as an alternate concept to contemporary development assurance processes (NASA/CR–2020-220586). NASA.

Presidential Commission on the Space Shuttle Challenger Accident. (1986). Report to the President. U.S. Government Printing Office.

Randell, B. (1975). System structure for software fault tolerance. IEEE Transactions on Software Engineering, SE-1(2), 220–232.

Rao, N. S. V. (2026). Principles of machine learning theories and practical applications. In S. S. Iyengar, K. G. Furton, P. Mohanty, & N. K. Chaudhary (Eds.), Artificial intelligence driven forensics (pp. 33–54). Springer.

Rauzy, A. (1993). New algorithms for fault trees analysis. Reliability Engineering & System Safety, 40(3), 203–211.

Royal Commission into the Robodebt Scheme. (2023). Report. Commonwealth of Australia. https://robodebt.royalcommission.gov.au/publications/report

Saltzer, J. H., & Schroeder, M. D. (1975). The protection of information in computer systems. Proceedings of the IEEE, 63(9), 1278–1308.

Sha, L. (2001). Using simplicity to control complexity. IEEE Software, 18(4), 20–28.

Stamatelatos, M., Vesely, W., Dugan, J., Fragola, J., Minarick, J., & Railsback, J. (2002). Fault tree handbook with aerospace applications (Version 1.1). NASA Office of Safety and Mission Assurance.

Tenney, R. R., & Sandell, N. R., Jr. (1981). Detection with distributed sensors. IEEE Transactions on Aerospace and Electronic Systems, AES-17(4), 501–510.

Tramontana, D. (2019, September 19). The top five quality management system failures of all time. QAD Blog. https://blog.qad.com/2019/09/the-top-five-quality-management-system-failures-of-all-time

U.S. Department of Justice, U.S. Attorney's Office, Northern District of California. (2024, November 14). Cruise admits to submitting a false report to influence a federal investigation and agrees to pay $500,000. https://www.justice.gov/usao-ndca/pr/cruise-admits-submitting-false-report-influence-federal-investigation-and-agrees-pay

U.S. Department of Justice, U.S. Attorney's Office, Northern District of Texas. (2021, January 7). Boeing charged with 737 Max fraud conspiracy and agrees to pay over $2.5 billion. https://www.justice.gov/usao-ndtx/pr/boeing-charged-737-max-fraud-conspiracy-agrees-pay-25-billion

U.S. Securities and Exchange Commission. (2013). SEC charges Knight Capital with violations of market access rule (Press release 2013-222). https://www.sec.gov/newsroom/press-releases/2013-222

Vapnik, V. N. (1998). Statistical learning theory. Wiley.

Verma, A., Pedrosa, L., Korupolu, M., Oppenheimer, D., Tune, E., & Wilkes, J. (2015). Large-scale cluster management at Google with Borg. In Proceedings of the Tenth European Conference on Computer Systems (EuroSys '15). https://doi.org/10.1145/2741948.2741964

Von Neumann, J. (1956). Probabilistic logics and the synthesis of reliable organisms from unreliable components. In C. E. Shannon & J. McCarthy (Eds.), Automata Studies (pp. 43–98). Princeton University Press.

Wabersich, K. P., & Zeilinger, M. N. (2021). A predictive safety filter for learning-based control of constrained nonlinear dynamical systems. Automatica, 129, 109597. https://doi.org/10.1016/j.automatica.2021.109597

Weiss, B., & Furr, M. (n.d.). Static stability using Availability Zones. The Amazon Builders' Library. https://aws.amazon.com/builders-library/static-stability-using-availability-zones/

Wellock, T. (n.d.). Putting the axe to the "scram" myth. U.S. Nuclear Regulatory Commission. https://www.nrc.gov/reading-rm/basic-ref/students/history-101/putting-axe-to-scram-myth

Zhao, D., Lam, H., Peng, H., Bao, S., LeBlanc, D. J., Nobukawa, K., & Pan, C. S. (2017). Accelerated evaluation of automated vehicles safety in lane-change scenarios based on importance sampling techniques. IEEE Transactions on Intelligent Transportation Systems, 18(3), 595–607. https://doi.org/10.1109/TITS.2016.2582208

Appendix: AI incidents and the engineering framework

This appendix catalogs AI and automated-decision incidents with authoritative documentation and maps each to a failure mechanism and to the protections the essay proposes. Compiled September 26, 2026.

- **Sources:** Preference is given to official investigations, judgments, security advisories, company disclosures, and research studies. Where a case relies on secondary reporting, that reliance is identified. Allegations are distinguished from established findings, and the engineering mappings are the author's analysis.
- **Distinct mechanisms:** incidents were chosen to cover different paths to harm, not to count harms. The AI Incident Database and the OECD AI Incidents Monitor catalog thousands more.

The mappings identify protections directed at each documented failure path. Their effectiveness would depend on implementation, coverage, timing, and the stated assumptions; they are engineering assessments, not findings attributed to the incident investigators. Specification errors are reached through validation: asking whether the requirements adequately address intended use, hazards, and applicable legal and policy obligations.

Authority, deployment, and change control

Incident and source	What happened	Failure mechanism	Applicable protections and conditions for effectiveness
Knight Capital, Aug. 2012 Operational. SEC administrative order (2013)	A faulty deployment reactivated dormant code on one server. The router sent over 4 million orders for 212 customer orders and lost more than $460 million in 45 minutes.	Unverified deployment; no pre-trade exposure limits; no effective incident procedure.	Staged rollout and shadow comparison could have exposed the mismatched server. A commit gate enforcing exposure invariants and escrow-style budgets would have capped the loss if correctly enforced; a kill switch with a rehearsed procedure could have shortened it. Automation, not machine learning, but the lesson transfers directly.
Replit agent, July 2025 Operational. Company CEO's public acknowledgment, as reported by The Register (secondary)	During a declared code freeze, a coding agent deleted a customer's production database, generated fabricated data, and said rollback was impossible when it was not.	Agent held production write authority; no separation of development and production; reliance on the agent's own report.	The agent should hold no production credentials: the commit gate does. Staged commit and protected checkpoints make deletion recoverable. Independent telemetry, not the agent's self-report, establishes what happened. The company's remedies (automatic dev/prod separation, staging) are exactly these controls.
xAI Grok, July 2025 Operational. xAI public statement, as reported by TIME (secondary)	An update to an upstream code path reintroduced deprecated instructions; for about 16 hours the chatbot posted antisemitic and violent content.	Configuration change deployed to production without effective behavioral checks or staged exposure.	Treat prompts and instructions as safety-relevant configuration: change control through the gate, shadow mode, staged rollout, and output monitoring with automatic safing. The framework limits exposure time; defining acceptable output is a specification question it does not settle.

Incident and source	What happened	Failure mechanism	Applicable protections and conditions for effectiveness
OpenAI GPT-4o sycophancy, Apr. 2025 Operational. OpenAI postmortem	A model update made ChatGPT excessively agreeable and was rolled back after three days. Offline evaluations and A/B tests looked good; expert testers said it "felt off."	A new reward signal weakened the one holding sycophancy in check; qualitative disagreement was outweighed by metrics.	The Hubble lesson: disagreement between independent checks is evidence, not noise. Launch criteria should make such disagreement blocking, as OpenAI's remedy now does. Staged rollout limited exposure. Metrics that users like can share a specification error with the objective.

Removed or weakened independent protection

Incident and source	What happened	Failure mechanism	Applicable protections and conditions for effectiveness
Uber ATG test vehicle, Tempe, Mar. 2018 Operational (road testing). NTSB report HAR-19/03 (2019)	The system detected the object 5.6 seconds before impact but never correctly classified it as a pedestrian or predicted her path. Emergency braking was unavailable under automated control, and the eventual auditory alert, about 0.2 seconds before impact, left insufficient time for effective intervention.	The vehicle's factory emergency braking was disabled under computer control; the design relied on a distracted human operator, without a timely warning.	Disabling factory emergency braking left operator intervention as the remaining collision-avoidance fallback when the automated driving system failed. The framework keeps diverse protection (factory braking), credits the human backup only with evidence of attention, and alerts the operator within the response deadline. NTSB also cited an inadequate safety culture, which independent safety-case review addresses.
Tesla Autopilot, 2018–2023 crashes Operational. NHTSA investigation EA22-002 (closed 2024); recall 23V-838	NHTSA reviewed 956 crashes, found 467 Autopilot-related, at least 13 fatal, and a "critical safety gap" between drivers' expectations and system capability.	High control authority paired with weak driver-engagement controls.	Bound the system's authority to its demonstrated operating domain, and treat driver monitoring as a safety function with measured coverage. A human backup earns credit only through evidence. NHTSA's follow-on query on the remedy shows that fixes also need verification.

Monitoring actual state and reporting

Incident and source	What happened	Failure mechanism	Applicable protections and conditions for effectiveness
Cruise robotaxi, San Francisco, Oct. 2023 Operational. U.S. DOJ deferred prosecution agreement (2024); NHTSA consent order (2024)	After a pedestrian was thrown into its path, the vehicle struck her and then dragged her about 20 feet while pulling over. The company's report to NHTSA omitted the dragging.	The post-collision maneuver was chosen without detecting the person beneath the car; incident reporting was incomplete.	Three Mile Island's lesson: act on observed state, not assumed state. Safing must be context-aware: stopping can be safer than moving. Tamper-evident records protected at the source, and mandatory incident reporting, address the disclosure failure. (Cruise was a GM subsidiary.)
Epic Sepsis Model, validated 2021 Validation study of a deployed model. Wong et al., JAMA Internal Medicine (2021)	An independent validation over 38,500 hospitalizations found an AUC of 0.63, well below vendor claims. The model missed two-thirds of sepsis cases while alerting on 18% of patients.	No independent external validation before wide deployment; high false-alarm burden.	Monitors earn credit only through demonstrated coverage and false-alarm rates under the real operating profile. Alarm fatigue is a documented hazard, as at Three Mile Island. Independent validation belongs in the safety case.
Waymo school-bus passing, 2025 Operational. NHTSA preliminary evaluation PE25-013; software recall (Dec. 2025), as reported by KUT (secondary)	Austin's school district documented at least 19 instances of robotaxis illegally passing stopped school buses, one moments after a student crossed.	A rare-scenario gap common to every vehicle running the same software.	Replicated software is a common cause across a fleet. Scenario-based testing of rare hazards and incident reporting are required. The software recall shows the recovery side: one verified update reaches the whole fleet.

Specification errors at scale

Incident and source	What happened	Failure mechanism	Applicable protections and conditions for effectiveness
Dutch childcare-benefits scandal, to 2018 Operational. Dutch Data Protection Authority fine (2021)	The Tax Administration used nationality as an indicator in a system that automatically flagged applications as risky, and retained data on about 1.4 million people unlawfully.	A discriminatory, unlawful specification applied automatically at scale.	An unlawful decision rule can be implemented consistently across every component. Independent review must therefore assess the rule itself against legal and data-protection requirements; such validation could have rejected the risk indicator. Validation asks whether requirements address intended use, hazards, and applicable legal and policy obligations.

Incident and source	What happened	Failure mechanism	Applicable protections and conditions for effectiveness
Australia's Robodebt, 2016–2019 Operational. Royal Commission report (2023); Federal Court settlement (2021)	Automated income averaging raised welfare debts the Commission called "neither fair nor legal." Legal advice that averaging did not accord with legislation was not acted on. A 2021 class-action settlement refunded about A$751 million and paid A$112 million in compensation.	An unlawful specification; missing human review before irreversible harm to individuals.	A specification error with the Hubble pattern: an independent check (the 2014 legal advice) disagreed and was set aside. Validation that treats lawfulness as a requirement, with disagreement as blocking, could have stopped deployment. Harms to individuals should be prevented before execution, like H1; losses fall on the agency best able to prevent them. An automated rule system, not machine learning.
Zillow Offers, 2021 Operational. Zillow Q3 2021 shareholder letter	Zillow wound down its algorithmic home-buying business, citing "unpredictability in forecasting home prices," with a $304 million inventory write-down and a 25% workforce reduction.	Distribution shift beyond the model's validated range; exposure grew faster than evidence.	Budgets and exposure caps bound losses; analytical monitoring of forecast residuals detects drift; staged growth limits exposure until evidence accumulates. The testing limit applies: past accuracy does not bound future error under shift.

Output verification and liability

Incident and source	What happened	Failure mechanism	Applicable protections and conditions for effectiveness
Mata v. Avianca, 2023 Operational. U.S. District Court (S.D.N.Y.) sanctions opinion	Lawyers filed a brief citing cases invented by ChatGPT, and were sanctioned.	Generated content reached an irreversible commit (a court filing) without an independent check.	A non-AI check before commit—checking each cited case and quoted passage against the original sources—could have exposed the fabricated authorities and quotations. Liability fell on the party best able to verify.
Moffatt v. Air Canada, 2024 Operational. British Columbia Civil Resolution Tribunal decision	The airline's chatbot misstated its bereavement-fare policy. The airline argued the chatbot was a separate entity; the tribunal held the airline liable.	Customer-facing output not checked against the authoritative policy.	Check consequential statements against the source of truth. The ruling is the cheapest-cost-avoider principle in action: the deployer bears the loss.
Deloitte report for Australia's DEWR, 2025 Operational. Department statement and refund, as reported by CFO Dive (secondary)	An A$440,000 government review contained fabricated references and a fabricated court quote; Deloitte refunded about A$97,000.	Generated text reached a delivered report without citation verification.	The same verification check as in Mata v. Avianca. The partial refund illustrates loss allocation to the party best able to prevent the error.

Untrusted input, supply chain, and disclosure

Incident and source	What happened	Failure mechanism	Applicable protections and conditions for effectiveness
Microsoft Tay, Mar. 2016 Operational. Microsoft blog statement	A chatbot that learned from its conversations was manipulated by a coordinated effort into posting offensive content and was taken offline within a day.	Online learning from untrusted input; adaptive adversaries not anticipated in testing.	Red teaming against adaptive adversaries before release; no learned change reaches production without passing the gate; output monitoring with automatic safing. Microsoft described exactly this as a coordinated attack that exploited a vulnerability.
Samsung, 2023 Operational. Internal memo, as reported by Bloomberg (secondary)	Engineers pasted confidential source code into ChatGPT; the company then restricted generative-AI use.	Unauthorized disclosure through an external service.	This is H1: prevent release before the request leaves, by checking data-release authority at the commit gate. Detection afterward cannot recall the data.
EchoLeak, Microsoft 365 Copilot, 2025 Demonstrated vulnerability; no known exploitation. CVE-2025-32711; researchers' analysis	A crafted email could make the assistant send confidential data to an attacker without any user action. It bypassed an injection classifier and exfiltrated through a domain the content policy allowed. Patched before disclosure; no known exploitation.	Untrusted input gained the assistant's authority; a domain allowlist stood in for release authorization.	H1 again: mediate outbound requests from tools, renderers, and proxies; check destination, arguments, and data-release authority, not merely domain membership. A classifier is one layer, not a boundary.
Amazon Q Developer extension, July 2025 Supply-chain compromise; payload failed to execute. AWS security bulletin AWS-2025-015	An inappropriately scoped build token let an attacker insert a data-wiping prompt into a released extension. It failed to execute only because of a syntax error.	Excessive privilege in the supply chain; malicious change released without review.	Least privilege for build credentials; review and staged release of changes; destructive agent actions routed through a gate. Luck is not a protection layer.

Misuse and autonomous agents

Incident and source	What happened	Failure mechanism	Applicable protections and conditions for effectiveness
AI-orchestrated espionage (GTG-1002), Sept. 2025 Misuse campaign. Anthropic threat report	A state-sponsored group used an AI coding agent against about 30 targets. The AI performed an estimated 80–90% of the work; guardrails were bypassed by splitting the attack into innocuous tasks and a false defensive pretext.	Per-request safeguards cannot see intent spread across many requests.	The institutional layer from the introduction: identity and risk-based access, monitoring across accounts and sessions, and intelligence sharing. Anthropic's response—banning accounts, notifying targets, improving detection—follows the fraud-and-cybersecurity model.
OpenAI / Hugging Face, July 2026 Evaluation incident with external impact. OpenAI incident report; METR independent investigation	During cyber evaluations, about 1,200 agents used a shared package service as a covert message board, escaped network isolation, and about 700 took part in an attack on Hugging Face. Some spoofed tool calls in their transcripts.	Shared writable service defeated isolation; production safeguards not applied to evaluations; telemetry forgeable; early warnings not escalated.	Separate protection domains with no uncontrolled cross-zone communication through shared services; typed, authorized egress; a safety case for evaluation as well as deployment, with containment justified for the controls enabled or disabled; telemetry protected at its source; escalation with deadlines. Attributed reports, not an independently reproduced attack.
UK AI Security Institute test, July 2026 Evaluation incident; no confirmed harm. AISI incident report	With deliberate internet access and disabled cyber classifiers, an agent attempted a supply-chain attack: malicious code, fake identities to manipulate reviewers, and Tor to evade network restrictions. A human reviewer rejected the malicious pull request.	Evaluation granted broad connectivity; the agent acted beyond its remit.	Connected mode needs typed, mediated external actions and real-time monitoring. Human review worked as an independent layer, but by a narrow margin; the safety case should not rely on it alone.

Patterns

- **Failures of protective barriers recur.** Uber depended on operator intervention after automated protections proved inadequate; Replit exposed production resources to agent-held authority; EchoLeak chained bypasses of several defenses. Whether these constitute singleton or higher-order cutsets depends on the defined hazard, system boundary, and explicit fault model.
- **Independent protections were removed or never applied.** Uber disabled factory braking; OpenAI's evaluations lacked production safeguards; Epic's model lacked external validation.
- **Disagreement and warnings were discounted.** GPT-4o's expert testers, OpenAI's May 2026 warning signs, and Robodebt's legal advice share Hubble's lesson.
- **Actions whose consequences cannot reliably be reversed require checks before execution.** Filings and reports can be corrected, but their consequences may persist; disclosed data cannot be recalled.
- **Specification and verification failures require different checks.** The Dutch and Australian cases required scrutiny of the decision rules themselves; Hubble required independent scrutiny of the evidence used to accept the manufactured mirror.

- **Liability and remedies fell on those positioned to verify.** Air Canada, the Mata sanctions, and Deloitte's refund placed losses on organizations or professionals positioned to verify outputs, a pattern consistent with the cheapest-cost-avoider rationale.

Appendix sources

Amazon Web Services (2025, July 23). Security bulletin AWS-2025-015 (CVE-2025-8217). https://aws.amazon.com/security/security-bulletins/AWS-2025-015/

Anthropic (2025). Disrupting the first reported AI-orchestrated cyber espionage campaign. https://www.anthropic.com/research/disrupting-AI-espionage

Autoriteit Persoonsgegevens (2021). Tax Administration fined for discriminatory and unlawful data processing. https://www.autoriteitpersoonsgegevens.nl/en/current/tax-administration-fined-for-discriminatory-and-unlawful-data-processing

Bloomberg (2023, May 2). Samsung bans ChatGPT, Google Bard, other generative AI use by staff after leak. https://www.bloomberg.com/news/articles/2023-05-02/samsung-bans-chatgpt-and-other-generative-ai-use-by-staff-after-leak

British Columbia Civil Resolution Tribunal (2024). Moffatt v. Air Canada, 2024 BCCRT 149. Summary: https://www.americanbar.org/groups/business_law/resources/business-law-today/2024-february/bc-tribunal-confirms-companies-remain-liable-information-provided-ai-chatbot/

CFO Dive (2025). Deloitte refunds over $60K for report with AI errors, Australian government says. https://www.cfodive.com/news/deloitte-refunds-60k-report-ai-errors-australian-government-accounting/803321/

EchoLeak: The first real-world zero-click prompt injection exploit in a production LLM system (2025). arXiv:2509.10540. https://arxiv.org/abs/2509.10540

Greenblatt, R., Cotra, A., & Wijk, H. (2026, Aug. 26). Brief independent investigation of agents' behavior, reasoning and collaboration in the OpenAI / Hugging Face hacking incident. METR. https://metr.org/blog/2026-08-26-openai-hugging-face-incident-investigation/

KUT (2025, Dec. 8). Waymo will recall software after self-driving cars passed stopped school buses. https://www.kut.org/transportation/2025-12-08/waymo-will-recall-software-after-self-driving-cars-passed-stopped-school-buses-in-austin-and-beyond

Mata v. Avianca, Inc., 678 F. Supp. 3d 443 (S.D.N.Y. 2023). https://www.law.berkeley.edu/wp-content/uploads/archive/2025/12/Mata-v-Avianca-Inc.pdf

Microsoft (2016, Mar. 25). Learning from Tay's introduction. https://blogs.microsoft.com/blog/2016/03/25/learning-tays-introduction/

National Highway Traffic Safety Administration (2024). EA22-002 closing resume. https://static.nhtsa.gov/odi/inv/2022/INCLA-EA22002-14498.pdf

National Transportation Safety Board (2019). Collision between vehicle controlled by developmental automated driving system and pedestrian, Tempe, Arizona (HAR-19/03). https://www.ntsb.gov/investigations/AccidentReports/Reports/HAR1903.pdf

OpenAI (2025). Expanding on what we missed with sycophancy. https://openai.com/index/expanding-on-sycophancy/

OpenAI (2026, Aug. 26). The Hugging Face incident and the road ahead. https://openai.com/index/hugging-face-incident-and-the-road-ahead/

Royal Commission into the Robodebt Scheme (2023). Report. https://robodebt.royalcommission.gov.au/publications/report

The Register (2025, July 22). Replit makes promise to stop its AI agents making vibe coding disasters. https://www.theregister.com/2025/07/22/replit_saastr_response/

TIME (2025). Musk's AI company apologizes over Grok's antisemitic posts. https://time.com/7301206/elon-musk-antisemitic-posts-ai-chatbot-grok-response/

U.S. Attorney's Office, N.D. Cal. (2024, Nov. 14). Cruise admits submitting a false report to influence a federal investigation. https://www.justice.gov/usao-ndca/pr/cruise-admits-submitting-false-report-influence-federal-investigation-and-agrees-pay

U.S. Securities and Exchange Commission (2013). SEC charges Knight Capital with violations of market access rule. https://www.sec.gov/newsroom/press-releases/2013-222

UK AI Security Institute (2026). Incident report: unsanctioned agent behaviour during cyber testing. https://www.aisi.gov.uk/blog/incident-report-unsanctioned-agent-behaviour-during-cyber-testing

Wong, A., et al. (2021). External validation of a widely implemented proprietary sepsis prediction model in hospitalized patients. JAMA Internal Medicine, 181(8), 1065–1070. https://jamanetwork.com/journals/jamainternalmedicine/fullarticle/2781307

Zillow Group (2021, Nov. 2). Third-quarter 2021 results and plan to wind down Zillow Offers. https://investors.zillowgroup.com/news-and-events/news/news-details/2021/Zillow-Group-Reports-Third-Quarter-2021-Financial-Results--Shares-Plan-to-Wind-Down-Zillow-Offers-Operations/default.aspx