iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** VIII **Month of publication:** August 2026

DOI: https://doi.org/10.22214/ijraset.2026.84530

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VIII Aug 2026- Available at www.ijraset.com

From Detection to Verified Action: Operational Readiness for AI-Enabled Cloud Failure Management

Adepegba Akindayomi Akintade
School of Computer Science and Information Systems, Osiri University, Nebraska, USA

Abstract: ***Artificial intelligence for information technology operations has progressed from alert correlation and anomaly detection to root-cause analysis, mitigation recommendation, and agents that can invoke infrastructure tools. This progression creates an assurance problem: analytical performance does not establish authority to change a live cloud system. This critical narrative review examines the evidence required before AI output may influence or execute failure-management action in mission-critical cloud infrastructure. Searches through 24 July 2026 covered scholarly databases, major systems venues, standards sources, and authoritative production reports. Forty-four sources were coded by operational task, evidence setting, authority, safety control, rollback, and outcome verification. Production evidence is substantial for detection, triage, diagnosis, and several narrowly bounded mitigation systems, but remains weak for general-purpose autonomous action. Recent agentic AIOps surveys emphasize contracts, bounded tools, canary deployment, and rollback; the unresolved need is an assessment method that separates what a system can infer, what it may do, and what evidence shows that the action remained safe. The proposed Operational Decision-Readiness and Verification framework represents a deployment claim through analytical capability, operational authority, and assurance maturity. Twelve domains use explicit 0-3 evidence anchors, while endpoint validity, evidence integrity, intervention risk, authorization, reversibility, and recovery verification operate as non-compensatory gates. Worked assessments of six systems illustrate distinctions among analytical support, testbed execution, and narrow production autonomy. The framework is conceptual and requires prospective and inter-rater validation; it structures an action-specific readiness case rather than certifying a model or product.***

Keywords: AIOps; cloud reliability; operational readiness; autonomous cloud; automated remediation; human oversight; rollback; recovery verification.

I. INTRODUCTION

Cloud services are operated through interdependent layers of physical hosts, networks, storage, virtualization, containers, orchestration, identity services, distributed applications, and continuously changing software. A local fault can be absorbed by redundancy, propagated through a dependency, or amplified by an attempted repair. Reliability therefore depends not only on detecting abnormal behaviour but also on interpreting service impact, selecting an appropriate response, controlling execution, and verifying recovery. This systems problem motivated autonomic computing and self-adaptive software long before contemporary large language models (Kephart and Chess, 2003; Salehie and Tahvildari, 2009; Schneider et al., 2015). The new development is that AI systems can now span much more of the operational workflow.

Artificial intelligence for information technology operations (AIOps) includes anomaly and incident detection, failure prediction, root-cause analysis, incident triage, mitigation recommendation, and automated remediation (Notaro et al., 2021; Cheng et al., 2023; Zhang et al., 2026). Large language models extend this portfolio by processing incident text, runbooks, code, configuration, and conversational queries. Tool-enabled agents can retrieve telemetry, execute diagnostic commands, prepare changes, and in controlled settings alter infrastructure. AIOpsLab was designed specifically to evaluate agents across the incident lifecycle in deployed cloud test environments with workloads, injected faults, telemetry, and orchestrated tool access (Chen et al., 2025), while related design work identifies the systems challenges of building agents for autonomous clouds (Shetty et al., 2024). These advances make limited autonomous operation technically credible.

The evidence needed to authorize action is nevertheless different from the evidence used to validate a model. A detector may achieve a high F1 score while generating an intolerable number of pages. A root-cause model may produce a plausible explanation without identifying the mechanism that determines the correct remedy.

A predictor may discriminate future failures but provide insufficient lead time. A mitigation recommender may overlook current topology, capacity, change windows, security policy, or rollback requirements. An agent may complete a benchmark task while remaining unsuitable for live authority because the test omits concurrent changes, misleading telemetry, partial execution, or irreversible effects.

Production systems illustrate both the promise and the narrowness of current closed-loop evidence. Warden improved cross-service incident detection through a global view of alerts but stopped at incident creation for human response (Li et al., 2021). RCACopilot used alert types and runtime diagnostic information to support cloud root-cause analysis, while engineers retained the operational decision (Chen et al., 2024). Language models evaluated on more than 40,000 incidents produced useful root-cause and mitigation suggestions, but usefulness did not establish execution safety (Ahmed et al., 2023). Narya predicted imminent Azure host failures, selected preventive mitigations, and reduced virtual-machine interruptions during production operation (Levy et al., 2020). RESIN detected, diagnosed, and automatically mitigated memory leaks over three years of production use (Lou et al., 2022). The latter systems show that verified automation is possible, but they succeed within tightly engineered failure classes and action spaces.

The security boundary has also changed. Operational agents consume evidence that can be incomplete, stale, or adversarial. Pasquini et al. (2026) demonstrated that manipulated telemetry can mislead LLM-driven AIOps agents into harmful infrastructure actions, showing that observability data are part of the attack surface rather than neutral ground truth. Tool-enabled agents also create risk through over-privileged tools, ambient authority, and mismatch between the user's intent and the capability actually exercised (Goel, 2026). Agent evaluation must therefore test evidence integrity, privilege, and trajectories, not only answer quality or task completion (Li et al., 2026).

Recent reviews have moved close to this control problem. Bilal et al. (2026) organize agentic NetOps and AIOps around autonomy rungs, tool scope, evidence traces, assurance contracts, non-bypassable gates, sandbox replay, canary trials, and rollback-aware evaluation. Their analysis makes a decisive contribution: operational reliability arises from the control machinery surrounding the model. The present review does not claim that contracts, bounded tools, or rollback are newly discovered concepts. Its narrower contribution is to convert these principles into an action-specific readiness assessment that distinguishes three questions commonly collapsed into one: what can the system infer, what may it execute, and what evidence supports the claimed level of assurance?

The review addresses four questions. First, which evidence gaps prevent successful detection, diagnosis, or prediction from justifying operational action? Second, how should analytical capability, action authority, and assurance maturity be represented without treating them as a single linear ladder? Third, which readiness domains must be assessed before a system is granted write authority? Fourth, how can the framework distinguish production analytical support, testbed execution, and narrow verified autonomy in published systems?

The paper makes four contributions. It first differentiates the proposed assessment from task-centred AIOps surveys and recent contract-centred agentic-operations surveys. It then introduces the Operational Decision-Readiness and Verification (ODRV) framework as a three-coordinate readiness state. Third, it provides twelve evidence domains with 0-3 rating anchors and identifies six non-compensatory gates for consequential actions. Finally, it applies the framework to six representative systems to demonstrate what the assessment can and cannot conclude.

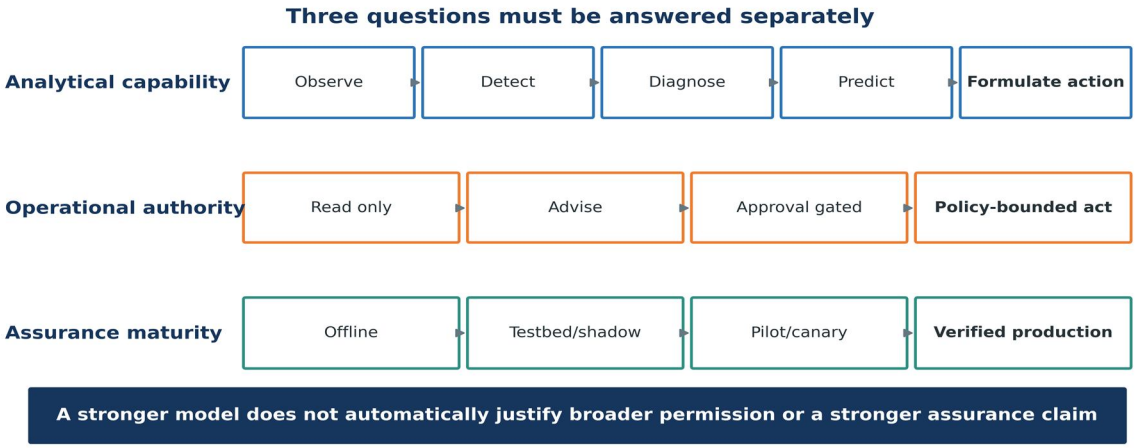


Fig. 1. Analytical capability, operational authority, and assurance maturity are separate questions. Progress in one dimension does not automatically justify progress in the others.

II. REVIEW APPROACH

A. Design and scope

A critical narrative design was used because the research problem crosses AIOps, autonomic and self-healing systems, software reliability, production machine learning, human-automation interaction, agent security, change safety, and AI governance. The purpose was conceptual integration rather than effect-size estimation or an exhaustive census. The review nonetheless used a documented search architecture and included-source register so that the basis of the synthesis is inspectable.

The principal evidence window was January 2020 through 24 July 2026. Earlier work was retained when it supplied a necessary concept, including dependable-computing terminology, autonomic control, calibration, dataset shift, concept drift, machine-learning production readiness, human interaction with automation, and safe rollout. The operational scope included public, private, and hybrid clouds; data centres; virtual machines; storage and network infrastructure; containerized and Kubernetes environments; microservices; and edge-cloud environments where infrastructure operation was central. Manufacturing prognostics, vehicle maintenance, ordinary help-desk automation, and cybersecurity detection without an infrastructure-action component were excluded.

B. Search architecture

Searches were conducted between 10 July and 24 July 2026 across IEEE Xplore, ACM Digital Library, Scopus, Web of Science, ScienceDirect, SpringerLink, arXiv, Google Scholar for citation tracing, USENIX proceedings, NIST publications, and official research pages where they provided a stable description of a production system or accepted paper. Search expressions were adapted to each interface and combined five blocks:

1. Infrastructure: cloud operations, data centre, virtual machine, Kubernetes, container, microservice, distributed system, network, storage. 2. Failure intelligence: AIOps, anomaly detection, incident detection, failure prediction, root-cause analysis, fault localization, predictive maintenance, self-healing. 3. Action: mitigation, automated remediation, restart, migration, scaling, isolation, failover, rollback, autonomous recovery. 4. Evidence quality: calibration, uncertainty, false alarm, prediction horizon, temporal validation, concept drift, prospective evaluation, service-level verification. 5. Control and security: authorization, least privilege, human oversight, safety constraint, canary deployment, audit trail, telemetry manipulation, tool security, post-deployment monitoring.

Representative combined expressions and the source register are supplied in the supplementary workbook. Raw hit counts are not reported because the review used heterogeneous database, venue, standards, and citation-tracing interfaces that do not expose stable comparable result sets. This limitation is preferable to presenting spurious precision. The final included corpus contains 44 sources; every source is identified in the register by publication class, operational role, evidence setting, and verification basis.

C. Inclusion, exclusion, and stopping rules

A source was retained when it met at least one of four conditions: it evaluated an AIOps or cloud-failure-management task; it described a production or testbed system connecting evidence to operational response; it supplied an independent safety mechanism such as canary analysis, rollback, policy validation, or monitoring; or it provided a foundational concept required to assess readiness. Commercial product descriptions without a disclosed method or evaluation were excluded. Preprints were retained only when they filled a recent evidence role and are labelled as such. When a peer-reviewed version was available, it was preferred.

Core sources were reviewed against structured fields covering endpoint, telemetry, validation design, deployment setting, action, human role, authority, rollback, service verification, calibration, lead time, false-action burden, security controls, and limitations. Contextual sources were used only for the concepts they directly support. Backward and forward citation tracing continued until two successive passes added no new readiness domain, evidence setting, or control mechanism. This is a saturation criterion for the conceptual codebook, not a claim that all AIOps publications were located.

D. Analytical Method

The original five-level maturity ladder was rejected during adversarial review because it mixed task stage, authority, and assurance. The revised analysis treats them as orthogonal dimensions. Capability records what the system can produce: observation and triage, detection, diagnosis, prediction, or an executable intervention plan. Authority records what the system is allowed to do: read-only analysis, advisory output, approval-gated execution, or policy-bounded autonomous execution. Assurance records where the claim has been tested: offline evaluation, testbed or shadow operation, limited pilot or canary, or longitudinal production verification.

Twelve readiness domains were derived through constant comparison across production AIOps systems, agent evaluation, safe deployment, rollback, human-automation, and governance literature. A source was never classified as highly assured merely because it used production data. Retrospective evaluation on production incidents remained offline evidence. Likewise, tool execution in a benchmark remained testbed evidence unless the study documented operational authority and verified service outcomes.

E. Method Limitations

The review is purposive, single-author, and not independently dual-screened. It may therefore reflect selection and interpretation bias. Industrial evidence is selectively disclosed, and unsuccessful deployments are under-reported. Several 2026 agentic-security sources are accepted papers or preprints rather than mature longitudinal studies. The ODRV ratings in Section 7 are illustrative applications based on published information, not independent audits of the systems. These constraints are retained explicitly because the framework itself should not be exempt from the evidence standards it proposes.

III. POSITIONING AND EVIDENCE LANDSCAPE

A. What earlier reviews establish

Notaro et al. (2021) organize 100 AIOps failure-management solutions by intervention window and target problem. Cheng et al. (2023) describe cloud AIOps through incident detection, failure prediction, root-cause analysis, and automated actions. Zhang et al. (2026) analyse 183 LLM4AIOps studies by data source, task, method, and evaluation. Remil et al. (2024) provide incident-management guidelines and a broad technical taxonomy. Reviews of proactive self-healing and microservice failure analysis add architectural and method-specific detail (Rouholamini et al., 2024; Schneider et al., 2015).

These reviews explain the problem space and the technologies used. Bilal et al. (2026) then advance the discussion from task taxonomies to constrained operational control, including tool scope, evidence traces, gates, canary execution, rollback, and audit. The present paper begins where that survey leaves an assessment problem: organizations need to make a defensible claim about a particular system, action, and environment. A contract specifies permitted behaviour; a readiness case evaluates whether the evidence justifies granting that permission.

Table 1. Positioning against prior AIOps and agentic-operations reviews.

Review line	Primary unit of analysis	Main contribution	Remaining need addressed by ODRV
Notaro et al. (2021)	AIOps failure-management task and intervention window	Taxonomy of 100 failure-management solutions	Does not separate capability, permission, and assurance for a specific action
Cheng et al. (2023)	Cloud AIOps data, tasks, and AI methods	Detection, prediction, RCA, and automated-action overview	Action authority and verified recovery remain secondary
Zhang et al. (2026)	LLM4AIOps data, tasks, methods, and evaluation	Survey of 183 studies in the LLM era	Evaluation is task-centred rather than an action-specific readiness case
Bilal et al. (2026)	Agentic NetOps/AIOps operational contract	Autonomy rungs, tool scope, evidence traces, gates, rollback	Needs a domain-rated assessment of whether a bounded claim is adequately evidenced
This review	Specific system, action, authority, and evidence setting	Three-coordinate readiness state, 12 domains, six gates, worked assessment	Requires prospective and inter-rater validation

B. The evidence gradient from observation to action

Published evidence is uneven across the operational lifecycle. Detection and triage have substantial production examples. Warden aggregates alerts across services and detects incidents at a global level (Li et al., 2021). COMET uses language models for interpretable incident triage and reported online deployment across cloud services (Wang et al., 2024). Diagnostic systems also use real incident corpora: RCACopilot combines incident routing, diagnostic aggregation, classification, and explanation (Chen et al., 2024); other work generates root causes and mitigation steps or retrieves diagnostic queries from operational context (Ahmed et al., 2023; Jiang et al., 2024; Zhang et al., 2024).

The evidence thins as systems approach write authority. A recommendation can be useful without being safe to execute. AIOpsLab provides a high-value test environment for end-to-end agents, but benchmark success is not production authorization (Chen et al., 2025). Narya and RESIN are stronger production cases because they connect a defined failure class to bounded action and outcome measurement (Levy et al., 2020; Lou et al., 2022). Their narrowness is not a defect; it is part of the assurance strategy. The operational semantics, action alternatives, and verification conditions are constrained enough to engineer and monitor.

Adjacent systems contribute controls that AIOps papers often omit. Google canary analysis limits exposure and chooses roll-forward, rollback, or escalation from observed production metrics (Beyer and Davidovic, 2018). Prodspec and Annealing implement intent-based incremental actuation with repeated validation (Palatin and Beyer, 2021). SafeRevert asks when evidence is sufficient to revert a breaking change automatically (Henderson et al., 2024). Safety constraints provide an independent barrier against dangerous automation (Schulman and Perot, 2018). These mechanisms demonstrate that safe action is a systems architecture, not a model feature.

C. A new security condition: the evidence may be hostile

Earlier AIOps discussions often treated telemetry quality as missingness, noise, delay, or drift. Agentic operation adds strategic manipulation. Pasquini et al. (2026) show that adversarial requests can inject plausible error interpretations into telemetry and steer an AIOps agent toward harmful changes. The study is important because it attacks the evidence-action chain itself. Sanitization, provenance checks, cross-signal corroboration, and tool-level policy cannot be deferred to a generic cybersecurity layer.

The same principle applies to tools. A cloud-hosted agent may be constrained in natural-language policy while still holding credentials or APIs with broader ambient authority. Goel (2026) identifies over-privileged tools, capability-intent mismatch, and authority leakage as central risks in tool-enabled execution environments. AgentCanary evaluates agent safety in executable environments using complete trajectories and separates outcome safety, security awareness, and task utility (Li et al., 2026). These studies support a stronger readiness requirement: evidence integrity and authority must be tested under adversarial as well as accidental failure.

D. Failure pathways, observability, and operational context

Cloud failure management begins with a measurement problem. A fault at one layer may appear as a symptom at another: storage latency can surface as application timeouts, packet loss can resemble overload, and a release defect can trigger resource exhaustion or retry amplification. The same telemetry value can also imply different risks under different workloads. High CPU utilization may be expected during batch processing, harmful for a latency-sensitive service, or evidence of a runaway process after a deployment. A readiness case must therefore bind evidence to topology, service criticality, redundancy, current capacity, recent change, and the intended service objective.

Observability supplies logs, metrics, traces, events, configuration, and dependency information, but these are not interchangeable. Logs can preserve semantic detail while lacking consistent structure. Metrics are efficient for trend and threshold detection but may omit causal context. Traces expose request paths and latency propagation but can be sampled or incomplete. Configuration and change records can identify the event that altered system behaviour, yet they may not capture undocumented operational work. The relevant evidence package should state which sources were available at decision time and which were reconstructed later.

The production incident literature shows that delay is often distributed across the lifecycle. Ghosh et al. (2022) found that detection, diagnosis, and mitigation problems interact in large cloud incidents. Parayil et al. (2023) similarly demonstrate that inadequate monitors and alert logic can delay recognition even where the service is nominally observable. This means a model should not be evaluated in isolation from the evidence pipeline and response workflow. A technically superior classifier can produce no operational benefit when data arrive late, ownership is unclear, or the recommended action cannot enter the change process.

Contextual retrieval can improve diagnosis. X-lifecycle learning combines code, configuration, monitoring, dependencies, and documents to support cloud incident management (Goel et al., 2024).

Query-recommendation systems lower the barrier to collecting discriminating operational evidence (Jiang et al., 2024). These methods also create governance obligations: every data source and tool call has a provenance, permission, latency, and failure mode. The broader the evidence surface, the greater the need to record what was queried, what was returned, and whether the result was current.

Endpoint and evidence quality should therefore be assessed together. A service-failure label supported only by a postmortem conclusion cannot be treated as a real-time prediction target unless the pre-incident evidence can be reconstructed without leakage. A root-cause explanation should identify the observations that distinguish it from alternatives. For high-consequence action, the evidence package should contain at least one independent corroborating signal or an explicit reason why corroboration is impossible.

IV. WHY ANALYTICAL PERFORMANCE DOES NOT ESTABLISH ACTION READINESS

A. Endpoint Validity

Dependable-computing terminology distinguishes a fault, an erroneous state, and a service failure (Avizienis et al., 2004). Operational datasets frequently collapse these distinctions. An anomaly may be harmless novelty. A component fault may be masked by redundancy. An incident label may reflect escalation practice rather than customer impact. A root-cause category may identify where symptoms were observed rather than the mechanism that should be changed. If the endpoint is weak, improved accuracy increases confidence in the wrong operational conclusion.

A readiness case must therefore specify the action-relevant endpoint, its timing, and its service relationship. AutoARTS, based on more than 2,000 postmortems across more than 450 Azure services, shows why a single root-cause label can be inadequate for multi-factor incidents (Dogga et al., 2023). A system that acts should represent contributing conditions, not force a complex incident into one convenient label.

B. Decision-time evidence and temporal realism

Operational evaluation must reproduce what was available when the decision would have been made. Random splitting can place observations from the same incident, host, or software regime in training and test sets. Features computed after escalation, mitigation, or postmortem review can leak the outcome. Chronological, entity-aware, and incident-aware separation is necessary when the claim is prospective. Lead time must be evaluated end to end. The relevant interval includes evidence collection, model inference, tool calls, review, change preparation, execution, and stabilization. A prediction delivered before a recorded failure can still be operationally late. Results should report warning-time distributions and the proportion of events detected within feasible action windows, not only average lead time.

C. Calibration, abstention, and false-action cost

Discrimination and calibration are different. Modern neural networks can be overconfident, and uncertainty can deteriorate under dataset shift (Guo et al., 2017; Ovadia et al., 2019). PACE-LM illustrates the value of confidence estimation for cloud root-cause analysis (Zhang et al., 2023). For action readiness, confidence must extend beyond the diagnosis to the intervention: probability that the endpoint is real, that the selected cause is relevant, that the action will help, and that unacceptable side effects will not occur.

False alerts consume attention; false actions alter the system. A low false-positive rate can still create high toil on a large event stream. An unnecessary restart, migration, or rollback consumes capacity and may induce a new incident. Evaluation should report unnecessary actions, harmful actions, avoided impact, and action-specific cost. Abstention is a valid result when evidence is insufficient or contradictory.

D. Generalization, drift, and recursive monitoring

Cloud environments change through releases, hardware refreshes, traffic shifts, topology changes, and new policies. Concept drift can change telemetry distributions and the mapping from evidence to outcome (Gama et al., 2014). Operational readiness therefore requires a monitoring plan for the model and agent as well as the infrastructure. NIST's analysis of deployed AI monitoring identifies fragmented terminology, degradation, logging barriers, and unresolved integration between automated and human validation (Rao et al., 2026). The monitoring loop is recursive. An AI system that monitors infrastructure requires its own health indicators, drift alarms, incident procedures, safe degraded mode, and version rollback. Hidden technical debt in data dependencies, thresholds, configurations, and consumers can undermine a model even when the algorithm is unchanged (Sculley et al., 2015; Breck et al., 2017; Amershi et al., 2019).

E. Explanation is not causal adequacy

Feature importance or fluent narrative may improve usability without identifying a cause that determines an effective intervention. Operational diagnosis should narrow the action space, represent alternatives, identify dependencies, state uncertainty, and predict an observable response to intervention. If the proposed cause is correct, a specified action should produce a specified change in service evidence. This creates a falsifiable remediation hypothesis.

Explanation stability also matters. If minor perturbations or missing signals produce different root causes and incompatible actions, the system may be unsuitable for write authority. Tool-using agents can collect discriminating evidence dynamically, but every additional tool expands the trust boundary (Roy et al., 2024). The system should escalate rather than improvise when explanations remain unstable.

F. Command completion is not recovery

An API success response proves only that a command was accepted or completed. Recovery is a service-level claim. It may require availability, latency, error rate, transaction completion, data integrity, dependency health, security posture, and capacity margin to remain within limits over a stabilization period. Immediate containment, minimum viable service, full restoration, and durable recovery are different states.

Rollback is likewise a capability with prerequisites, not a generic command. The prior state must be recoverable; data and configuration must remain compatible; the trigger must be defined; and rollback itself must be verified. Some actions are only partially reversible. The evidence threshold should increase with irreversibility and blast radius.

Table 2. Evidence required beyond analytical performance.

Dimension	Minimum evidence	Failure if omitted
Endpoint validity	Action-relevant event, label provenance, timing, service-impact relationship	An anomaly or component event is treated as a service failure
Decision-time evidence	Only evidence available when the decision would occur; chronological and entity-aware validation	Retrospective leakage inflates apparent prediction or diagnosis
Lead time	Distribution of warning time after evidence collection and decision latency	Correct output arrives too late to support the action
Calibration and abstention	Reliability under shift; action-conditioned thresholds; insufficient-evidence state	High confidence is mistaken for high reliability
False-action burden	Unnecessary and harmful actions, induced incidents, operational cost	A low false-positive rate produces unacceptable change volume
Diagnostic adequacy	Alternatives, dependencies, uncertainty, and falsifiable post-action expectation	A plausible narrative is treated as a causal mechanism
Drift and recursive monitoring	Model/agent health, drift response, safe degraded mode, version rollback	The automation silently degrades after deployment
Service verification	Service objectives, side effects, stabilization, recurrence, rollback outcome	Command completion is reported as recovery

V. THE REVISED ODRV FRAMEWORK

A. A readiness claim is a tuple, not a ladder

The revised Operational Decision-Readiness and Verification framework represents a deployment claim as:

$R = (C, A, S \mid D, X)$

where C is analytical capability, A is operational authority, S is assurance maturity, D is the twelve-domain evidence profile, and X is the specific action and environment. The final term is essential. A system may be ready to restart a stateless replica under a tested policy but not ready to modify a database schema, repair data, or reconfigure a regional control plane.

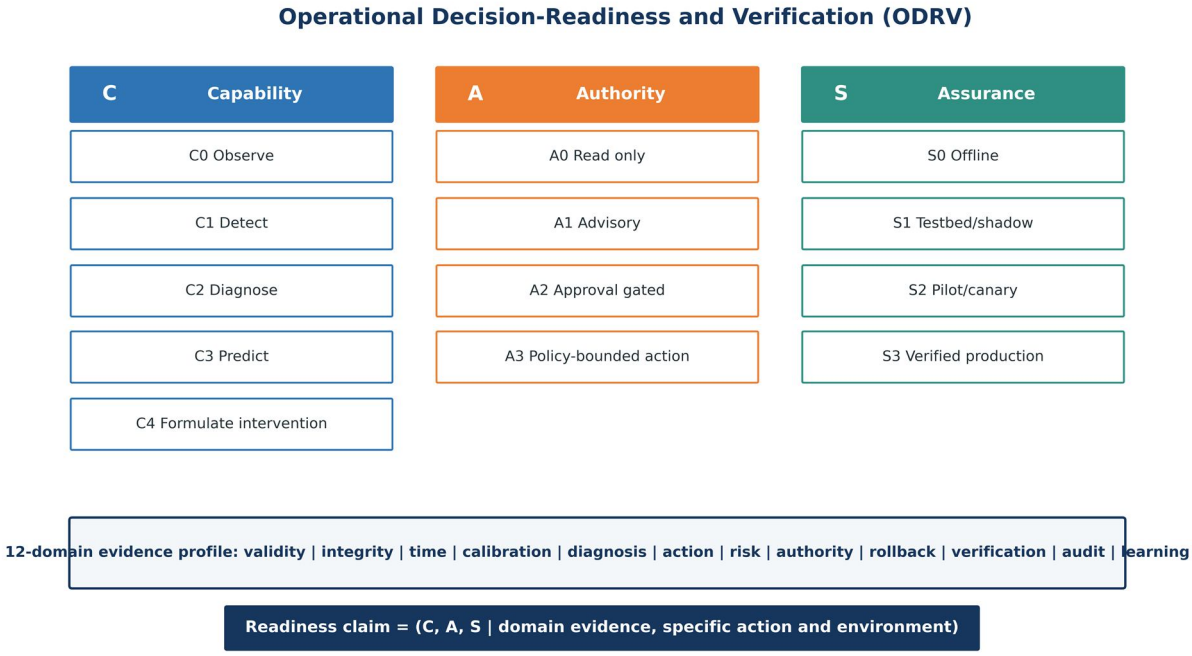


Fig. 2. Revised ODRV framework. A readiness claim is represented by analytical capability (C), operational authority (A), assurance maturity (S), a twelve-domain evidence profile, and the specific action and environment.

B. Capability Coordinate

Capability records what the system can produce independently of permission:

- C0 - Observe and summarize: retrieve evidence, correlate context, summarize state. - C1 - Detect and triage: identify abnormal or incident conditions and route response. - C2 - Diagnose: rank or explain plausible contributing causes. - C3 - Predict: estimate a future failure, degradation, or incident with an action window. - C4 - Formulate intervention: produce an executable, state-aware intervention contract.

A higher capability does not imply broader authority. A C4 system can remain read-only, and a deterministic C1 controller can be allowed to execute a narrow restart policy.

C. Authority Coordinate

Authority records the permitted operational role:

- A0 - Read-only: observe, retrieve, and summarize without proposing a change. - A1 - Advisory: recommend investigation or action; a human independently executes. - A2 - Approval-gated execution: the system prepares and may execute after an explicit policy or human approval gate. - A3 - Policy-bounded autonomous execution: the system acts without case-by-case approval inside a pre-authorized action envelope, with stop and override mechanisms.

Authority is determined by organizational policy, credentials, tool design, and action envelope, not by model architecture. Least privilege and separation of duties are technical properties of this coordinate.

D. Assurance coordinate

Assurance records where and how the claim has been evaluated:

- S0 - Offline: retrospective or static benchmark evidence only. - S1 - Testbed or shadow: execution in a representative test environment, replay, or non-actuating shadow deployment. - S2 - Limited operational exposure: pilot, canary, constrained tenant or region, explicit rollback, and monitored guardrails. - S3 - Longitudinal production verification: repeated production operation with measured service outcomes, negative outcomes, monitoring, and change control.

AIOpsLab can support high capability and tool interaction at S1; it does not by itself confer S3. Narya and RESIN support narrow S3 claims because production operation and outcomes are reported.

E. Twelve Readiness Domains

The twelve domains are grouped into four functions:

1. Evidence validity: endpoint validity, telemetry adequacy and integrity, temporal validity and lead time. 2. Decision quality: calibration and uncertainty, diagnostic adequacy, action specificity. 3. Execution safety: intervention risk, authorization boundary, rollback and reversibility. 4. Assurance and learning: recovery verification, auditability, post-action monitoring and learning.

Each domain is rated on a four-point evidence anchor:

- 0 - absent: not addressed or not reported; - 1 - described: requirement or mechanism is described but not evaluated; - 2 - controlled evidence: evaluated offline, in replay, testbed, or limited controlled exposure; - 3 - operationally verified: evaluated in the claimed environment with relevant outcomes and documented limitations.

The rating describes the published evidence, not the intrinsic quality of a system. An undisclosed control must be scored as not evidenced, even if it may exist internally.

F. Non-compensatory gates

A summed score can conceal a critical failure. Consequential actions therefore require six gates that cannot be offset by strength elsewhere:

1. Endpoint gate: the action-relevant event and service impact are valid. 2. Evidence-integrity gate: telemetry provenance, freshness, and adversarial integrity are adequate. 3. Intervention-risk gate: benefit, no-action alternative, dependencies, and blast radius are evaluated. 4. Authorization gate: permission, separation of duties, stop, and override are explicit. 5. Reversibility gate: rollback or an alternative safe-containment plan is tested where technically possible. 6. Verification gate: service recovery and unacceptable side effects are measured.

For an A3 claim, all applicable gates should be rated at least 2 before limited exposure and 3 before a general S3 claim. This is a proposed decision rule requiring validation, not a universal standard.

Table 3. ODRV readiness domains, rating anchors, and non-compensatory gates.

Domain	Core decision question	0-3 evidence anchor	Gate?
Endpoint validity	Is the target the action-relevant operational event?	0 absent; 1 defined; 2 controlled validation; 3 operationally verified	Yes
Telemetry adequacy and integrity	Is evidence timely, complete, traceable, and resistant to manipulation?	0 absent; 1 described; 2 tested; 3 verified under operational/adversarial conditions	Yes
Temporal validity and lead time	Is the complete decision path early enough for feasible action?	0 absent; 1 estimated; 2 controlled timing; 3 operational timing distribution	No
Calibration and uncertainty	Does stated confidence match observed reliability and action risk?	0 absent; 1 confidence shown; 2 calibrated in validation; 3 monitored under shift	No
Diagnostic adequacy	Does the diagnosis distinguish alternatives and support a falsifiable action?	0 absent; 1 plausible; 2 controlled evidence; 3 operational confirmation	No
Action specificity	Is there an executable state-aware intervention contract?	0 absent; 1 generic advice; 2 executable in test; 3 operationally verified	No
Intervention risk	Are benefit, no-action, dependencies, and blast radius evaluated?	0 absent; 1 described; 2 controlled assessment; 3 operational outcome evidence	Yes
Authorization boundary	Are tools, credentials, approvals, stop, and override explicit?	0 absent; 1 policy described; 2 enforced in test/pilot; 3 verified in production	Yes
Rollback and reversibility	Can action be halted or reversed and the reversal verified?	0 absent; 1 plan; 2 tested; 3 operationally exercised	Yes where applicable
Recovery verification	Did the service recover without unacceptable side effects?	0 absent; 1 proxy; 2 controlled service verification; 3 longitudinal operational verification	Yes
Auditability	Can evidence, versions, decisions, and tool calls be reconstructed?	0 absent; 1 partial log; 2 complete controlled trace; 3 operational trace governance	No
Post-action learning	Are actual effects used to revise model, policy, and thresholds?	0 absent; 1 review described; 2 structured controlled learning; 3 longitudinal feedback loop	No

G. Risk-linked authority

The required evidence depends on consequence and reversibility. Read-only summarization can tolerate more uncertainty than data repair. A low-blast-radius restart of a stateless replica may be eligible for A3 after repeated S2 validation. Regional traffic reconfiguration may require A2 or a narrowly bounded A3 policy. Destructive or weakly reversible operations should remain A1 or A2 unless independent assurance is exceptionally strong.

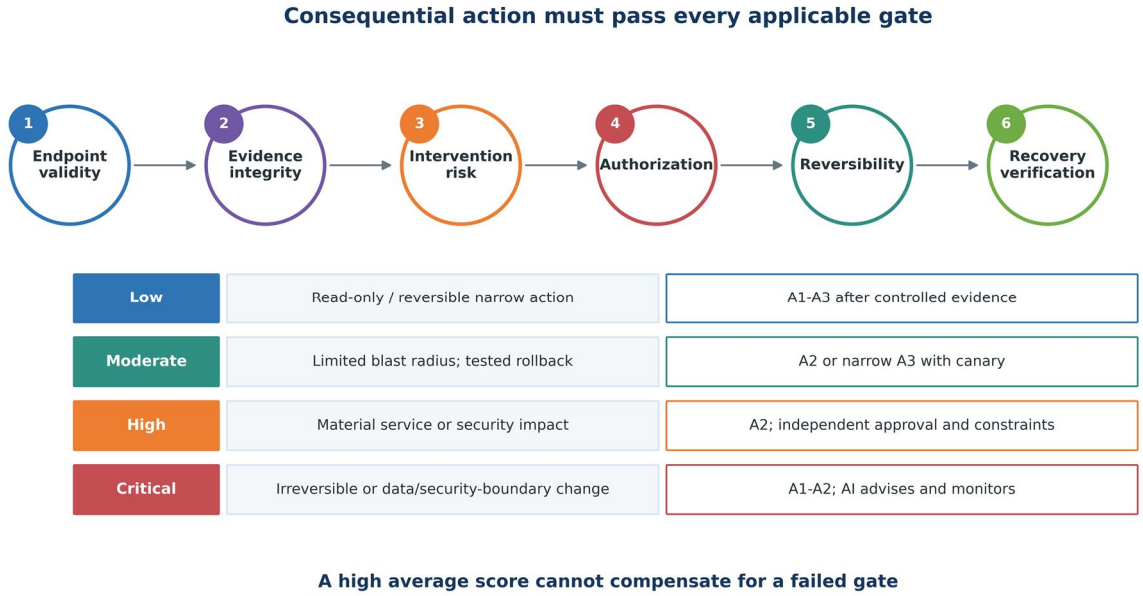


Fig. 3. Non-compensatory gates and risk-linked authority. A consequential action must pass every applicable gate; strength in another domain cannot offset a failed safety condition.

H. Using ODRV as a readiness case

An assessment begins with one proposed action, one environment, and one intended authority. The assessor records the C-A-S coordinates, rates the twelve domains, applies the six gates, and states the residual uncertainty. The output is a profile and decision rationale rather than a certification label. Reassessment is required after changes to the model, telemetry, tools, permissions, infrastructure, or action policy.

The framework also prevents vendor-level generalization. Evidence that a product safely restarts one class of service does not show readiness to manage identity, storage, or data-repair operations. Readiness is attached to a bounded operational claim.

I. Decision procedure and reporting template

A practical ODRV assessment can be completed in seven steps. First, define the exact operational claim: the action, target resource, environment, incident class, and requested authority. Second, identify the current capability coordinate and prohibit claims beyond demonstrated output. Third, document the actual tool permissions and approval path rather than the intended policy alone. Fourth, assign the assurance coordinate from the strongest relevant evidence setting. Fifth, rate the twelve domains with a citation or operational record for every score above zero. Sixth, apply the non-compensatory gates and record any condition that blocks the requested authority. Seventh, state the decision, residual uncertainty, monitoring plan, and trigger for reassessment.

The report should contain both the proposed and permitted state. For example, a team may propose C4-A3-S2 for automated traffic shifting, but the assessment may permit only C4-A2-S2 because rollback has controlled evidence while evidence-integrity testing remains descriptive. This distinction prevents a development roadmap from being reported as achieved readiness.

ODRV also supports conditional decisions. Authority may be granted only for a defined region, service class, time window, or capacity state. It may be suspended automatically when telemetry integrity fails, drift exceeds a threshold, concurrent change is detected, or rollback becomes unavailable. These conditions should be machine-enforceable where possible and included in the action log.

VI. SAFE ACTION ARCHITECTURE

A. The Intervention Contract

A C4 output should be an intervention contract rather than a free-standing command. The contract specifies the target condition, evidence, current uncertainty, expected benefit, affected resources, prerequisites, parameters, success criteria, stop conditions, rollback, and accountable authority. It also records the no-action alternative. Action selection should be ranked by expected end-to-end service effect, not only by local symptom removal (Namyar et al., 2025). This structure makes the operational hypothesis testable.

For example, a host-failure prediction may justify migration only when destination capacity, storage consistency, network reachability, and tenant constraints are satisfied. The success criterion is not that the migration API returns success. It is that workload availability and performance remain within limits and no unacceptable secondary pressure occurs.

B. Independent validation and constrained tools

The component proposing an action should not be the only component deciding that the action is safe. Policy engines, invariants, topology checks, capacity checks, and security controls provide independent validation. CloudCanary illustrates pre-change analysis of correlated failure risk (Zhai et al., 2020).

Data-centre safety constraints show why automation should be treated as a potential hazard requiring an independent barrier (Schulman and Perot, 2018).

Tool permissions should be typed by effect. Log queries, diagnostic scripts, configuration proposals, workload migration, and destructive repair are not equivalent tool calls. Read tools should be separated from proposal tools and execution tools. Credentials should be short-lived and scoped to the action envelope. Goel's (2026) analysis of privileged execution environments reinforces that policy text is insufficient when ambient authority remains broader than the declared task.

C. Adversarial evidence checks

Evidence validation should include provenance, schema, freshness, cross-source consistency, and manipulation testing. The AIOpsDoom attack demonstrates that plausible telemetry content can steer agents toward harmful actions (Pasquini et al., 2026). Defensive design should therefore sanitize user-influenced telemetry, distinguish data-plane symptoms from control instructions, corroborate high-consequence decisions across independent evidence, and prevent telemetry text from changing the agent's governing policy.

Red-team evaluation should test indirect prompt injection, poisoned runbooks, stale topology, forged alerts, missing signals, permission errors, and partial tool failure. AgentCanary's trajectory-grounded approach is relevant because unsafe behaviour may appear across a sequence rather than in the final answer (Li et al., 2026).

D. Staged execution, rollback, and verification

Execution should progress through dry run, sandbox or replay, shadow operation, canary exposure, and expansion only when the action permits such staging.

Automated canary analysis provides a mature pattern: compare a limited exposure with a suitable baseline, monitor guardrails, and choose expand, hold, rollback, or escalate (Beyer and Davidovic, 2018). Prodspec and Annealing similarly repeat selection, update, and validation under intent constraints (Palatin and Beyer, 2021).

Rollback requires a recoverable state, trigger, execution path, and verification. SafeRevert provides a direct precedent for treating automatic reversion as a sufficient-evidence decision (Henderson et al., 2024). When rollback is impossible, the action envelope should narrow and the approval burden increase.

Verification must return to service objectives. Guardrails can include availability, latency, error rate, transaction completion, data integrity, security posture, dependency health, and capacity margin. The system should distinguish immediate containment, minimum viable service, full restoration, and durable recovery. Post-action observation should continue for a defined window and compare predicted with actual effects.

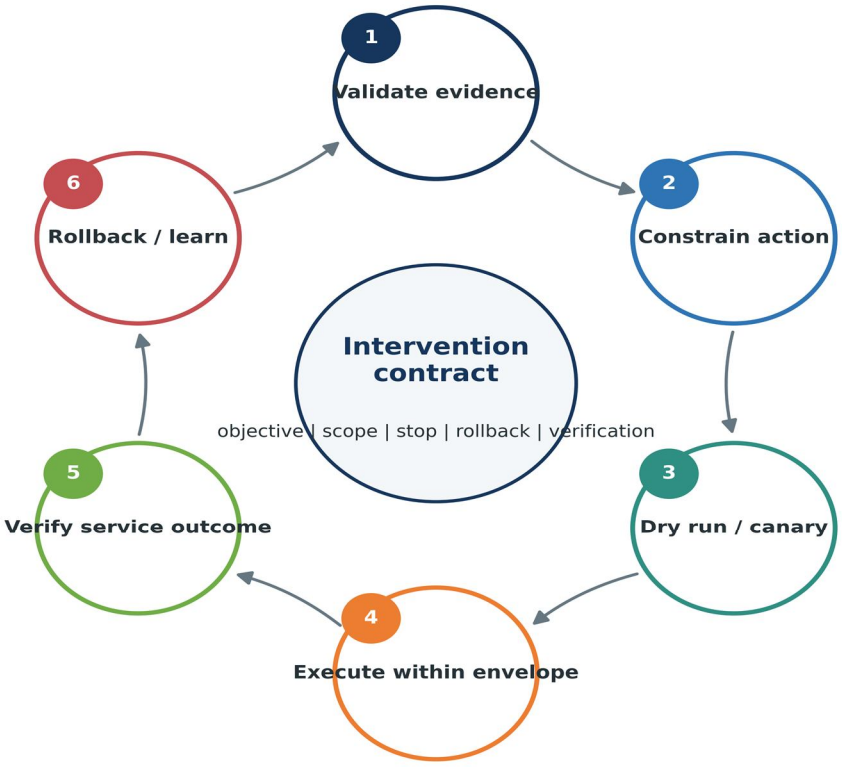


Fig. 4. Safe action cycle. The intervention contract links evidence, constrained execution, service verification, rollback, and learning.

E. Human authority and meaningful oversight

Human oversight is not meaningful merely because a person can click approve. Automation can change workload, attention, and the complexity of verification. Human-automation research distinguishes levels of automation and documents automation bias and verification burden (Parasuraman et al., 2000; Lyell and Coiera, 2017). An approval gate is weak when the operator lacks time, evidence, expertise, or an independent way to challenge the recommendation. Meaningful A2 control requires a concise evidence trace, alternatives, uncertainty, expected effect, blast radius, and rollback status. High-risk changes may require separation between diagnosis, approval, and execution. Operators should be able to stop the agent, revoke credentials, freeze further action, and recover manually. Training and simulation should include failure of the automation itself.

VII. WORKED ODRV ASSESSMENTS

The following assessments use published evidence only. They are not independent audits and should not be interpreted as certification. Their purpose is to demonstrate how the three coordinates prevent unlike systems from being placed on a single maturity ladder.

A. Warden

Warden is a production incident detector that aggregates cross-service alerts to improve global detection (Li et al., 2021). Its published claim is C1-A1-S3: production detection with human operational response. Endpoint, telemetry, lead time, and alert burden receive meaningful evidence. Action specificity, authorization for remediation, rollback, and recovery verification are not applicable to the detector's actual role. The system should not be described as low maturity because it does not act; it is highly assured for a narrower analytical role.

B. RCACopilot

RCACopilot combines incident-handler matching, diagnostic information, classification, and explanatory generation for cloud incidents (Chen et al., 2024). Its claim is approximately C2-A1-S2: diagnostic support using production incident evidence with human decision authority. Diagnostic adequacy and auditability are stronger than in free-form generation, but the reported accuracy and recommendation setting do not justify write authority. Calibration, action risk, and recovery verification remain separate requirements.

C. AIOpsLab

AIOpsLab exposes agents to deployed test environments, injected faults, telemetry, and tools across the incident lifecycle (Chen et al., 2025). An evaluated agent may reach C4-A2-S1 within the laboratory: it can investigate and execute through an orchestrator, but the assurance claim remains testbed based. AIOpsLab is therefore a platform for building the evidence needed for S2, not evidence that a general agent is production ready.

D. Narya

Narya combines imminent host-failure prediction, adaptive mitigation selection, online experimentation, and measured reduction in virtual-machine interruptions (Levy et al., 2020). For the reported host-failure use case, it supports C3/C4-A3-S3. The narrow endpoint and bounded mitigation set strengthen action specificity and outcome verification. The publication does not justify extending the claim to unrelated failure classes or unconstrained agent tools.

E. RESIN

RESIN detects, diagnoses, and mitigates production memory leaks using targeted evidence and automatic intervention (Lou et al., 2022). It supports C2/C4-A3-S3 for a narrow memory-leak domain. Its architecture demonstrates why decomposition can be safer than general end-to-end autonomy. The ODRV claim is strong for the documented failure and action, not for cloud management in general.

F. Automated canary control

Google's canary-analysis service is not an AIOps failure-prediction model, but it is an important assurance control for operational action (Beyer and Davidovic, 2018). It supports C4-A3-S3 for rollout assessment: evaluate a proposed change under limited exposure, then roll forward, roll back, or escalate. As an adjacent system, it shows that write authority can be highly assured even when the analytical component is statistical and narrow. The relevant achievement is the controlled decision architecture.

Table 4. Illustrative ODRV assessments based on published evidence.

System	ODRV coordinates	Strongest published evidence	Readiness inference
Warden	C1-A1-S3	Production cross-service detection, alert aggregation, incident creation	Strong production detector; no remediation-authority claim is needed
RCACopilot	C2-A1-S2	Production incident context, diagnostic pipeline, human decision role	Diagnostic support; published evidence does not justify write authority
AIOpsLab agents	C4-A2-S1	Deployed test environments, injected faults, telemetry, orchestrated tool access	End-to-end test capability; not production assurance
Narya	C3/C4-A3-S3	Bounded host-failure prediction, adaptive mitigation, measured interruption reduction	Narrow verified production autonomy for the reported use case
RESIN	C2/C4-A3-S3	Three-year production detection, diagnosis, mitigation, and reboot outcome	Narrow verified autonomy for memory-leak management
Canary analysis service	C4-A3-S3	Limited exposure, metric comparison, expand/rollback/escalate decision	Mature assurance control; shows safe actuation need not be general AI

The worked examples reveal three implications. First, capability and authority are genuinely independent. Warden can be a successful production system without remediation authority. Second, testbed execution and production execution are not equivalent: AIOpsLab can exercise broad workflows at S1 while Narya and RESIN support narrower S3 claims. Third, high readiness often depends on constraining the problem rather than maximizing model generality.

G. Cross-case Synthesis

The six cases show why a single total maturity score would be misleading. Warden has strong production assurance but intentionally limited capability and authority. AIOpsLab exposes richer agent behaviour but remains a testbed. Narya and RESIN support autonomous production claims because the failure classes, actions, and outcome measures are bounded. Canary analysis supplies assurance for change execution even though it is not a general AIOps model.

The comparison also reveals that the weakest domain changes with the claim. For an incident detector, recovery verification is outside scope and should not be treated as a deficiency. For a system requesting A3 authority, the same absence is disqualifying. ODRV therefore evaluates fitness for a proposed role, not universal sophistication. This role-specific interpretation is the main practical difference between the framework and a linear autonomy ladder.

VIII. RESEARCH AND IMPLEMENTATION PRIORITIES

A. Validate the Assessment Instrument

ODRV requires prospective validation. Multiple assessors should independently rate the same systems, and agreement should be measured. Domain anchors should be refined where disagreement is high. Prospective studies should test whether stronger profiles predict fewer harmful actions, faster recovery, or more reliable escalation. Without such work, ODRV remains a structured argument rather than a validated measurement instrument.

B. Build intervention-aware datasets and benchmarks

Most datasets stop at detection or diagnosis. Useful readiness benchmarks should include decision-time evidence, candidate actions, permissions, execution traces, rollback, side effects, and service outcomes. Compound failures, stale state, concurrent changes, and tool failure should be included. Evaluation should score the complete trajectory, consistent with emerging agent-safety frameworks (Li et al., 2026).

C. Link calibration to consequence

Probability thresholds should be action specific. The cost of a missed low-risk restart differs from the cost of an unnecessary data repair. Research should estimate action-conditioned utility, abstention, and uncertainty decomposition. A single global confidence score is inadequate for systems that move across different tools and blast radii.

D. Secure the evidence-action chain

AIOps security research should treat telemetry, retrieval stores, runbooks, tool descriptions, and control APIs as a connected attack surface. Defences should combine provenance, sanitization, least privilege, independent policy, trajectory monitoring, and post-action verification. Telemetry manipulation should become a standard red-team scenario, not an exceptional threat model (Pasquini et al., 2026).

E. Evaluate human-AI teams and manual recovery

Studies should measure whether oversight improves decisions under realistic time pressure. Relevant outcomes include verification quality, automation bias, response latency, workload, trust calibration, and ability to recover when the agent fails. Nominal human approval should not be counted as a safeguard without evidence that the operator can exercise it meaningfully.

F. Maintain living assurance cases

A readiness case should record the action, environment, coordinates, domain evidence, gates, residual risks, and monitoring plan. It should be revised after changes to data, model, tools, credentials, topology, software, or policy. NIST's AI Risk Management Framework supports lifecycle governance, monitoring, incident response, recovery, and change management (Tabassi, 2023). ODRV translates those principles into a cloud-failure-management assessment structure.

G. A staged implementation pathway

Organizations adopting AI-enabled failure management should separate capability development from authority expansion. The first stage is a read-only evidence assistant evaluated for retrieval accuracy, provenance, and operator usefulness. The second stage is shadow decision support: the system proposes diagnoses or interventions while humans act independently, allowing calibration and disagreement analysis. The third stage is approval-gated execution for low-blast-radius actions with dry-run checks and tested rollback. The fourth stage is policy-bounded autonomy under canary exposure. Only after repeated service-level verification and negative-outcome review should the action progress to broader S3 use.

Every stage should have an exit criterion and a regression criterion. Expansion is justified by evidence, not elapsed time. Regression is required when the infrastructure, model, tools, permissions, or failure distribution changes materially. A living ODRV profile can therefore function as a change-control artifact connecting model development, SRE, security, and governance teams.

Table 5. Research and implementation priorities.

Priority	Required advance	Expected value
Instrument validation	Inter-rater testing, prospective studies, anchor refinement	Determines whether ODRV ratings are reliable and predictive
Intervention-aware benchmarks	Decision-time evidence, actions, permissions, trajectories, rollback, outcomes	Moves evaluation beyond classification and final-answer accuracy
Action-conditioned calibration	Thresholds and abstention linked to reversibility and harm	Reduces confident but unsafe action
Evidence-action security	Telemetry provenance, sanitization, least privilege, trajectory red teaming	Protects the control loop from manipulation and authority leakage
Human-AI team evaluation	Verification quality, workload, automation bias, manual recovery	Tests whether oversight is meaningful
Living assurance cases	Reassess after model, tool, permission, topology, or policy change	Keeps authority aligned with current evidence

IX. LIMITATIONS AND IMPLICATIONS

This review does not establish that ODRV predicts safe deployment. The framework was derived from literature and tested through worked applications, not prospective field evaluation. The ratings were made by one author and may not be reproducible without further calibration. The included corpus is purposive and cannot support claims about publication prevalence. Industrial reporting is selective, and several important controls may be omitted from public papers. Conversely, published success in one environment may not transfer to another.

The fast-moving 2026 evidence also requires caution. Bilal et al. (2026), Goel (2026), and Li et al. (2026) are preprints; Pasquini et al. (2026) is an accepted forthcoming conference paper. They are included because they directly affect the novelty and security analysis, but stable conclusions continue to rely on peer-reviewed systems work, production reports, and established safety mechanisms.

For practitioners, the framework's principal implication is restraint. A high-performing model should not be assigned broader permissions merely because it is more capable. The proposed operational claim should be decomposed into capability, authority, assurance, and a domain profile. For researchers, the framework highlights a shift from model-centred evaluation to action-centred evidence. For reviewers and regulators, it provides questions that expose when a paper or product conflates offline accuracy, recommendation usefulness, testbed execution, and verified operational benefit.

X. CONCLUSION

AI-enabled cloud failure management is moving toward agents that can investigate incidents and invoke operational tools. The decisive question is no longer whether such systems can act, but when the evidence justifies allowing a particular action in a particular environment. Existing AIOps reviews define tasks and methods, and recent agentic-operations surveys correctly emphasize contracts, bounded tools, gates, canary execution, and rollback. The remaining assessment problem is to separate analytical capability from permission and assurance.

The revised ODRV framework represents readiness as an action-specific tuple of capability, authority, assurance, and twelve evidence domains. Six non-compensatory gates prevent strong performance in one area from masking a critical failure in endpoint validity, evidence integrity, intervention risk, authorization, reversibility, or recovery verification. Worked examples show that production detection, testbed execution, and narrow autonomous mitigation are different achievements and should be reported as such.

ODRV is not a certification standard and should not be used as a simple total score. Its value is to make the readiness claim explicit, falsifiable, and revisable. The next step is prospective and inter-rater validation across organizations and failure classes. Until that evidence exists, the safest principle remains clear: model capability may support an operational decision, but only systems evidence can justify operational authority.

A. *Declarations*

1) Funding: No external funding was received for this study.

2) Competing interests: The author declares no competing interests.

3) Data and materials availability: The review protocol, final included-source register, evidence coding, ODRV rating guide, search log, and worked assessment matrix are provided as supplementary materials. No proprietary operational data were used.

4) Author contributions: Adepegba Akindayomi Akintade: Conceptualization, methodology, investigation, evidence synthesis, framework development, writing, visualization, and final approval.

REFERENCES

[1] Ahmed, T., Ghosh, S., Bansal, C., Zimmermann, T., Zhang, X., and Rajmohan, S. (2023). Recommending root-cause and mitigation steps for cloud incidents using large language models. In Proceedings of the 45th IEEE/ACM International Conference on Software Engineering, 1737-1749. https://doi.org/10.1109/ICSE48619.2023.00149

[2] Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., and Zimmermann, T. (2019). Software engineering for machine learning: A case study. In Proceedings of the 41st International Conference on Software Engineering: Software Engineering in Practice, 291-300. https://doi.org/10.1109/ICSE-SEIP.2019.00042

[3] Avizienis, A., Laprie, J.-C., Randell, B., and Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. IEEE Transactions on Dependable and Secure Computing, 1(1), 11-33. https://doi.org/10.1109/TDSC.2004.2

[4] Beyer, B., and Davidovic, S. (2018). Canary analysis service. ACM Queue, 16(5), 70-95. https://doi.org/10.1145/3291278.3291290

[5] Bilal, M., Crowcroft, J., Wang, R., Xu, X., and Dustdar, S. (2026). Large language models for agentic NetOps and AIOps: Architectures, evaluation, and safety. arXiv:2605.12729. https://doi.org/10.48550/arXiv.2605.12729

[6] Breck, E., Cai, S., Nielsen, E., Salib, M., and Sculley, D. (2017). The ML test score: A rubric for ML production readiness and technical debt reduction. In 2017 IEEE International Conference on Big Data, 1123-1132. https://doi.org/10.1109/BigData.2017.8258038

[7] Chen, Y., Shetty, M., Somashekar, G., Ma, M., Simmhan, Y., Mace, J., Bansal, C., Wang, R., and Rajmohan, S. (2025). AIOpsLab: A holistic framework to evaluate AI agents for enabling autonomous clouds. Proceedings of Machine Learning and Systems, 7. https://arxiv.org/abs/2501.06706

[8] Chen, Y., Yang, H., Zhang, C., Ma, M., Kang, Y., Li, Z., He, S., Zhang, X., Wang, R., Rajmohan, S., Lin, Q., and Zhang, D. (2024). Automatic root cause analysis via large language models for cloud incidents. In Proceedings of the Nineteenth European Conference on Computer Systems, 674-688. https://doi.org/10.1145/3627703.3650083

[9] Cheng, Q., Sahoo, D., Saha, A., Yang, W., Liu, C., Woo, G., Singh, M., Savarese, S., and Hoi, S. C. H. (2023). AI for IT operations on cloud platforms: Reviews, opportunities and challenges. arXiv:2304.04661. https://doi.org/10.48550/arXiv.2304.04661

[10] Dogga, P., Parayil, A., Ghosh, S., Bansal, C., Zhang, X., Mace, J., Nath, S., and Rajmohan, S. (2023). AutoARTS: Taxonomy, insights and tools for root cause labelling of incidents in Microsoft Azure. In 2023 USENIX Annual Technical Conference, 359-375. USENIX Association.

[11] Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. (2014). A survey on concept drift adaptation. ACM Computing Surveys, 46(4), Article 44. https://doi.org/10.1145/2523813

[12] Ghosh, S., Shetty, M., Bansal, C., and Nath, S. (2022). How to fight production incidents? An empirical study on a large-scale cloud service. In Proceedings of the 13th Symposium on Cloud Computing, 126-141. https://doi.org/10.1145/3542929.3563472

[13] Goel, D., Husain, F., Singh, A., Ghosh, S., Parayil, A., Bansal, C., Zhang, X., and Rajmohan, S. (2024). X-lifecycle learning for cloud incident management using large language models. In Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering, 417-428. https://doi.org/10.1145/3663529.3663861

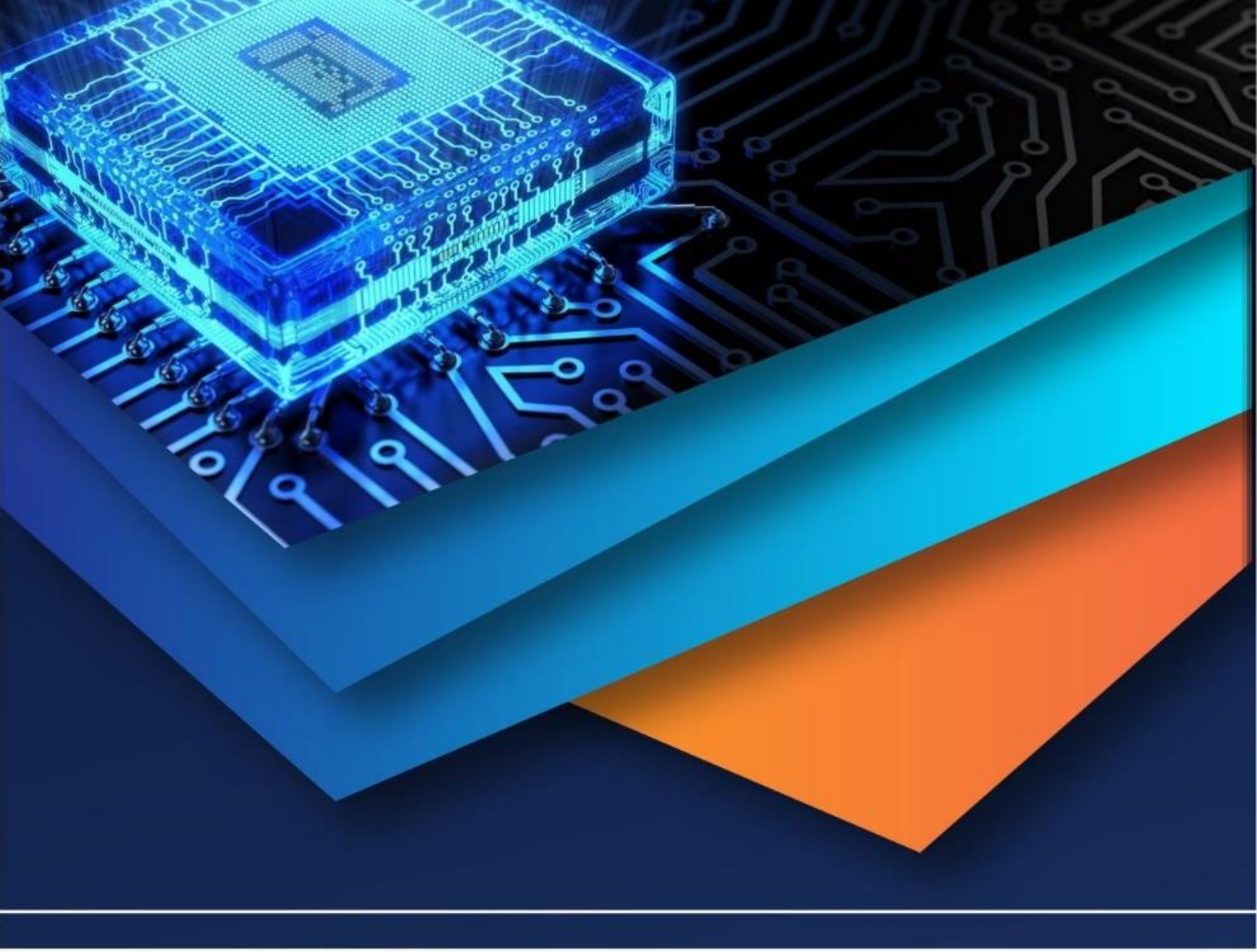
[14] Goel, H. (2026). Security risks in tool-enabled AI agents: A systematic analysis of privileged execution environments. arXiv:2605.09721. https://doi.org/10.48550/arXiv.2605.09721

[15] Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In Proceedings of the 34th International Conference on Machine Learning, PMLR 70, 1321-1330.

[16] Henderson, T., Kondareddy, A., Azad, S., and Nickell, E. (2024). SafeRevert: When can breaking changes be automatically reverted? In 2024 IEEE Conference on Software Testing, Verification and Validation. IEEE.

[17] Jiang, Y., Zhang, C., He, S., Yang, Z., Ma, M., Qin, S., Kang, Y., Dang, Y., Rajmohan, S., Lin, Q., and Zhang, D. (2024). Xpert: Empowering incident management with query recommendations via large language models. In Proceedings of the 46th IEEE/ACM International Conference on Software Engineering, Article 92, 1-13. https://doi.org/10.1145/3597503.3639081

[18] Kephart, J. O., and Chess, D. M. (2003). The vision of autonomic computing. Computer, 36(1), 41-50. https://doi.org/10.1109/MC.2003.1160055
[19] Levy, S., Yao, R., Wu, Y., Dang, Y., Huang, P., Mu, Z., Zhao, P., Ramani, T., Govindaraju, N., Li, X., Lin, Q., Shafriri, G. L., and Chintalapati, M. (2020). Predictive and adaptive failure mitigation to avert production cloud VM interruptions. In 14th USENIX Symposium on Operating Systems Design and Implementation, 1155-1170. USENIX Association.
[20] Li, L., Zhang, X., Zhao, X., Zhang, H., Kang, Y., Zhao, P., Qiao, B., He, S., Lee, P., Sun, J., Gao, F., Yang, L., Lin, Q., Rajmohan, S., Xu, Z., and Zhang, D. (2021). Fighting the fog of war: Automated incident detection for cloud systems. In 2021 USENIX Annual Technical Conference, 131-146. USENIX Association.
[21] Li, P., Wang, S., Huang, Y., Shi, Y., Zhang, C., Li, Q., Lyu, Y., Shan, C., Li, F., Feng, C., Zhu, C., and Chen, L. (2026). AgentCanary: A security evaluation framework for autonomous AI agents in real executable environments. arXiv:2606.10484. https://doi.org/10.48550/arXiv.2606.10484
[22] Lou, C., Chen, C., Huang, P., Dang, Y., Qin, S., Yang, X., Li, X., Lin, Q., and Chintalapati, M. (2022). RESIN: A holistic service for dealing with memory leaks in production cloud infrastructure. In 16th USENIX Symposium on Operating Systems Design and Implementation, 109-125. USENIX Association.
[23] Lyell, D., and Coiera, E. (2017). Automation bias and verification complexity: A systematic review. Journal of the American Medical Informatics Association, 24(2), 423-431. https://doi.org/10.1093/jamia/ocw105
[24] Namyar, P., Ghavidel, A., Crankshaw, D., Berger, D. S., Hsieh, K., Kandula, S., Govindan, R., and Arzani, B. (2025). Enhancing network failure mitigation with performance-aware ranking. In 22nd USENIX Symposium on Networked Systems Design and Implementation, 335-357. USENIX Association.
[25] Notaro, P., Cardoso, J., and Gerndt, M. (2021). A survey of AIOps methods for failure management. ACM Transactions on Intelligent Systems and Technology, 12(6), Article 81. https://doi.org/10.1145/3483424
[26] Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., and Snoek, J. (2019). Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. Advances in Neural Information Processing Systems, 32.
[27] Palatin, T., and Beyer, B. (2021). Prodspec and Annealing: Intent-based actuation in Google production. USENIX ;login:.
[28] Parasuraman, R., Sheridan, T. B., and Wickens, C. D. (2000). A model for types and levels of human interaction with automation. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, 30(3), 286-297. https://doi.org/10.1109/3468.844354
[29] Pasquini, D., Kornaropoulos, E. M., Ateniese, G., Akgul, O., Theocharis, A., and Efstathopoulos, P. (2026). When AIOps become "AI Oops": Subverting LLM-driven IT operations via telemetry manipulation. 35th USENIX Security Symposium, accepted prepublication paper. https://www.usenix.org/conference/usenixsecurity26/presentation/pasquini
[30] Rao, A., Keller, A., Kalra, N., Steed, R., Kwegyir-Aggrey, K., Klyman, K., Staheli, D., and Bergman, A. (2026). Challenges to the monitoring of deployed AI systems. NIST AI 800-4. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.AI.800-4
[31] Remil, Y., Bendimerad, A., Mathonat, R., and Kaytoue, M. (2024). AIOps solutions for incident management: Technical guidelines and a comprehensive literature review. arXiv:2404.01363. https://doi.org/10.48550/arXiv.2404.01363
[32] Rouholamini, S. R., Mirabi, M., Farazkish, R., and Sahafi, A. (2024). Proactive self-healing techniques for cloud computing: A systematic review. Concurrency and Computation: Practice and Experience, 36(24), e8246. https://doi.org/10.1002/cpe.8246
[33] Roy, D., Zhang, X., Bhave, R., Bansal, C., Las-Casas, P., Fonseca, R., and Rajmohan, S. (2024). Exploring LLM-based agents for root cause analysis. In Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering, 208-219. https://doi.org/10.1145/3663529.3663841
[34] Salehie, M., and Tahvildari, L. (2009). Self-adaptive software: Landscape and research challenges. ACM Transactions on Autonomous and Adaptive Systems, 4(2), Article 14. https://doi.org/10.1145/1516533.1516538
[35] Schneider, C., Barker, A. D., and Dobson, S. A. (2015). A survey of self-healing systems frameworks. Software: Practice and Experience, 45(10), 1375-1398. https://doi.org/10.1002/spe.2250
[36] Schulman, C., and Perot, E. (2018). Help protect your datacenters with safety constraints. Google Research.
[37] Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., and Dennison, D. (2015). Hidden technical debt in machine learning systems. Advances in Neural Information Processing Systems, 28.
[38] Shetty, M., Chen, Y., Somashekar, G., Ma, M., Simmhan, Y., Zhang, X., Mace, J., Las-Casas, P., Gupta, S., Nath, S., Bansal, C., and Rajmohan, S. (2024). Building AI agents for autonomous clouds: Challenges and design principles. In Proceedings of the 15th ACM Symposium on Cloud Computing.
[39] Tabassi, E. (2023). Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.AI.100-1
[40] Wang, Z., Li, J., Ma, M., Li, Z., Kang, Y., Zhang, C., Bansal, C., Chintalapati, M., Rajmohan, S., Lin, Q., Zhang, D., Pei, C., and Xie, G. (2024). Large language models can provide accurate and interpretable incident triage. In 2024 IEEE 35th International Symposium on Software Reliability Engineering, 523-534. https://doi.org/10.1109/ISSRE62328.2024.00056
[41] Zhang, L., Jia, T., Jia, M., Wu, Y., Liu, A., Yang, Y., Wu, Z., Hu, X., Yu, P. S., and Li, Y. (2026). A survey of AIOps in the era of large language models. ACM Computing Surveys, 58(2), Article 44. https://doi.org/10.1145/3746635
[42] Zhang, D., Zhang, X., Bansal, C., Las-Casas, P. H. B., Fonseca, R., and Rajmohan, S. (2023). PACE-LM: Prompting and augmentation for calibrated confidence estimation with GPT-4 in cloud incident root cause analysis. arXiv:2309.05833. https://doi.org/10.48550/arXiv.2309.05833
[43] Zhang, X., Ghosh, S., Bansal, C., Wang, R., Ma, M., Kang, Y., and Rajmohan, S. (2024). Automated root causing of cloud incidents using in-context learning with GPT-4. In Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering, 266-277. https://doi.org/10.1145/3663529.3663846
[44] Zhai, E., Chen, A., Piskac, R., Balakrishnan, M., Tian, B., Song, B., and Zhang, H. (2020). Check before you change: Preventing correlated failures in service updates. In 17th USENIX Symposium on Networked Systems Design and Implementation, 575-589. USENIX Association.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY