International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

Design and Evaluation of a Safety-Gated Cloud Security Posture Management System with Episode-Level Evidence Provenance

Nihal Ranjan Gowda*1, Kiran Kumar MN*2
*1UG Scholar, Department of Computer Application, BMS College of Commerce and Management, Bengaluru, Karnataka, India.
*2HOD, Department of Computer Application, BMS College of Commerce and Management, Bengaluru, Karnataka, India.

Abstract. Automated correction of cloud misconfigurations must distinguish authorization to act, verified resource state, and historical evidence retrieved afterward. This paper evaluates an AWS cloud security posture management prototype integrating two-stage authorization, pre-mutation revalidation, episode ownership, postcondition checks, retrospective exposure investigation, and version-specific audit delivery. Three playbooks address IAM access keys, role trust, and Secrets Manager resource policies. Controlled experiments recorded 90/90 verified remediation trials, 24/24 expected safety and failure-control outcomes, nine successful re-violation pairs comprising 18 correction episodes, and 18/18 exposure windows. Investigation matched nine seeded positive events by CloudTrail event identity and returned nine scope-qualified negative findings. Separate runner checks passed for 36 retained S3 object versions. The results demonstrate correction and evidence linkage on controlled fixtures; they do not establish robustness to delayed telemetry or safety for live workload dependencies.

Keywords: cloud security posture management; automated remediation; AWS; safety gates; postcondition verification; evidence provenance; CloudTrail

1 Introduction

Cloud configuration changes can broaden role trust, leave an access key active, or grant secret access to an unapproved principal. A control is a configured rule for acceptable resource state; a violation is failure to satisfy that rule. A playbook implements its detection and correction, while a safety gate checks eligibility to act. An episode is the durable tracking unit for a detected violation of one control on one resource. It groups observations, remediation decisions, and linked evidence assessments under an episode ID. A new violation after closure starts a new episode. Successful correction alone establishes neither authorization correctness nor absence of earlier activity.

AWS Config, Cloud Custodian, and AWS Automated Security Response already provide policy-driven correction [1–3]. The question here is how a bounded implementation can keep mutation eligibility, postconditions, retrospective activity, and audit retention separately inspectable.

Four research questions map directly to the experiments: RQ1 (E1), whether scoped playbooks repeatedly restore intended AWS state; RQ2 (E2), whether exclusions, pre-mutation state changes, held ownership, and denied permissions produce expected decisions; RQ3 (E3), whether reintroduced violations receive new correction episodes; and RQ4 (E4), whether investigation matches seeded activity, returns scoped negative findings, and verifies linked audit versions.

This study contributes a concrete AWS implementation and controlled evaluation linking postcondition-verified remediation episodes to bounded CloudTrail exposure reconstruction, scope-qualified non-observation, and independently checked audit versions.

2 Related Work and Positioning

2.1 Managed cloud remediation and policy-as-code

AWS Config exposes remediation actions, execution status, and retry controls [1]. Cloud Custodian applies filters and actions, including event-triggered operation [2]. AWS Automated Security Response integrates Security Hub findings with manual or automatic playbooks, resource filtering, and retry bounds [3]. These systems establish policy-driven correction and guardrails as existing capabilities.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

Prowler emphasizes broad posture detection and remediation guidance [4]. This prototype evaluates three adapters, the control-specific modules that read resource state, detect violations, repair supported cases, and verify results.

2.2 IAM policy semantics and verification

Graph-based IAM analysis detects anomalous relationships [5], while AWS formal reasoning and Access Analyzer provide stronger semantic analysis than handcrafted predicates [6, 7]. Here, a predicate is the implemented test for an accepted resource state. Verification means a fresh AWS read satisfies that test. Unsupported policy forms require operator review.

2.3 Evidence integrity and provenance

CloudTrail history supports bounded regional management-event lookup [8]; log-file integrity validation uses signed digests [9]. S3 GOVERNANCE Object Lock restricts deletion of particular versions but permits authorized bypass [10]. These mechanisms support different assertions. The prototype therefore separates recorded activity, query coverage, and audit retention rather than treating a retained report as proof of complete historical visibility.

2.4 Forensic investigation and remediation evidence

Daubner et al. integrate risk-based forensic readiness into SensitiveCloud and assess investigation support [11]. CIAF automates cloud-log interpretation through ontology validation and LLM analysis using Azure ransomware logs [12]. These works establish evidence availability and forensic automation as prior art; this implementation instead links deterministic, control-specific historical queries to correction episodes.

CUSTOS acquires volatile state before destructive containment [13]. Here, an asynchronous worker retrieves already recorded AWS events after correction; it cannot recover destroyed volatile evidence. CIAF and CUSTOS are cited as preprints. These different sequencing choices address different evidence sources.

Agade and Balpande describe policy-gated AIOps with evidence packages for each remediation run, including policy traces, provenance, signatures, and post-action validation [14]. Their evaluation uses simulated incident episodes. Our evaluation exercises live AWS fixtures and checks historical CloudTrail event identity, qualified negative exposure windows, and exact retained S3 versions. The contribution concerns this implementation and evaluation scope, without claiming first use of per-remediation evidence or comparative superiority.

3 System Model and Architecture

3.1 Deployment scope, actors, and assumptions

Terraform provisions one AWS account with regional functions and stores in us-east-1; IAM is account-global. CloudTrail/EventBridge trigger three Python 3.14/Boto3 adapters. DynamoDB stores episodes, SQS carries investigation jobs, and S3 retains audit versions. CloudWatch, SNS, recovery functions, a six-hour scanner, and an authenticated read-oriented dashboard support operation.

Mutation means changing AWS resource configuration. Mutation scope is the operator-authorized target set, distinct from the historical query scope. Adapters assess live state; the evidence worker interprets history; dashboard readers observe results. Events trigger fresh reads rather than establish authoritative state. The model covers misconfiguration, duplicate/stale delivery, concurrency, denied writes, and incomplete history. Compromised root authority and actors able to bypass retention controls are outside scope.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

3.2 Architectural separation of live correction and historical claims

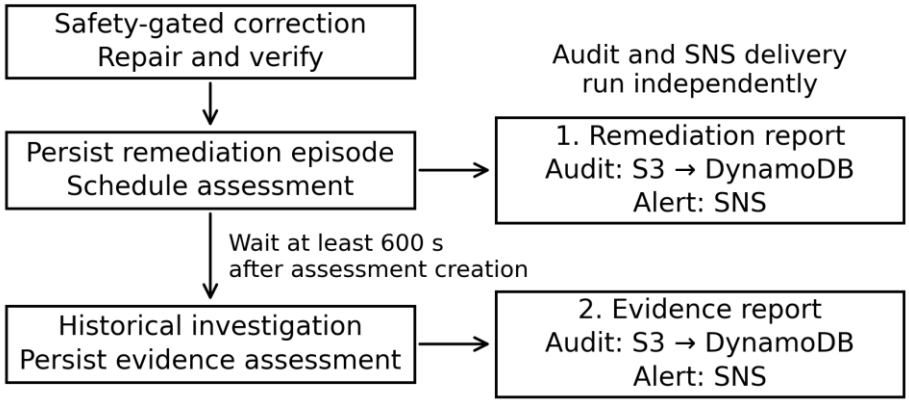


Fig. 1. Remediation and evidence produce separate reports. Each creates independent audit and SNS jobs. Audit delivery writes S3, then the DynamoDB audit table used by the dashboard.

Adapters authorize, claim an episode, re-read before mutation, and check the postcondition: the required resource state after correction. FIXED denotes a correction verified by read-back, a fresh AWS API read. Finalization atomically commits the episode, remediation delivery jobs, and assessment scheduling to DynamoDB. These audit and notification jobs are immediately eligible; investigation waits at least 600 s after assessment creation. This normally allows the remediation alert to publish first; the processes retry independently. Finalized evidence creates a second audit/notification pair.

3.3 Record identities and claim boundaries

A finding identifies a control–resource pair across recurrences; an episode identifies one tracked occurrence. An observation is an event-triggered or scanner-triggered encounter with that resource. An assessment is a separately versioned historical investigation linked to the episode. Provenance means the recorded links among observations, decisions, assessments, and retained objects. A mutation receipt records the action and verification metadata; an audit receipt identifies an S3 version, submitted-byte digest, and retention metadata. Several observations and records can belong to one episode; these are not independent trials.

4 Safety-Gated Remediation Implementation

4.1 Implemented security controls

Table 1. Three playbooks and their control-specific postconditions.

Control	**Recognized unsafe state and gate**	**Corrective API / postcondition**
IAM access key	Active key on a scoped user without an MFA device; never-touch, opt-out, and dependency-review gates.	UpdateAccessKey(Inactive); ListAccessKeys confirms the same key is inactive.
IAM role trust	Trust allows a principal outside the approved set or an unsupported unsafe relationship; preserve approved relationships.	UpdateAssumeRolePolicy; new trust document satisfies the supported predicate.
Secrets Manager policy	Unsafe/public resource-policy relationships; deletion only when explicitly authorized.	PutResourcePolicy or approved DeleteResourcePolicy; policy read-back is compliant.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

All adapters share scope checks, episode recording, revalidation, and read-back. Empty scope does not authorize mutation. The key predicate is an active key on a user without an MFA device, subject to dependency review. This is an operator-defined policy: MFA enrollment does not enforce MFA on ordinary access-key requests, and the predicate alone does not establish that deactivation is operationally safe. Role/secret edits remove unapproved principals from supported unconditional, exact-action Allows. Condition, Not*, ambiguous principals, and unsafe broad/mixed edits require review. Removing every role Allow requires review; an empty secret replacement requires explicit deletion approval.

4.2 Two-stage authorization and TOCTOU checks

Stage A excludes never-touch identities, approved exceptions, and out-of-scope targets before resource reads. Stage B checks live tags, key metadata, trust, and resource policies. After episode acquisition, a second read and gate evaluation precede mutation. A security-state digest is a hash of normalized read state; a changed digest blocks the write. This addresses time-of-check to time-of-use (TOCTOU) changes between inspection and action. The invocation uses its loaded configuration snapshot. The final read narrows but cannot eliminate the remaining read-to-write race.

4.3 Concurrency, recovery, and decision status

DynamoDB conditional transactions coordinate ownership [15]. A claim grants one invocation a time-limited ownership lease for the episode; competing invocations cannot claim the same held lease. The outbox is durable pending delivery work committed with its result. Recovery revisits interrupted work; reconciliation checks the recorded action and live state before deciding how to finish it. Decision, evidence, and delivery recovery are separate. DynamoDB atomicity does not extend across AWS resource APIs.

RESOLVED_EXTERNALLY denotes compliance observed without a verified engine repair; it receives no automatic-repair credit. Exclusions, review/escalation, and failed or unverified writes retain separate outcomes.

4.4 Testing hooks and production equivalence

A synchronization hook is a test-only pause allowing a controlled resource change before the final re-read. E2 uses this default-disabled, fixture-scoped hook to exercise TOCTOU behavior. Offline controls exercise the actual engine with simulated dependencies and no live AWS mutation. They support tested control flows rather than every live interleaving.

5 Exposure Investigation, Audit, and Operator Presentation

5.1 Retrospective exposure reconstruction

An exposure window is the historical interval queried for activity while a resource may have violated the control. Its onset is the reconstructed transition into violation; if unknown, a bounded search interval is used. The recorded correction closes a resolved window. Initial assessment eligibility is 600 s after creation, allowing historical events time to become queryable. Early SQS messages are rescheduled. A separate delivery wait runs from window closure: Terraform sets 600 s; the worker adds DELIVERY_DELAY until this interval expires. The fallback is 900 s only when the setting is absent. These waits can overlap and do not automatically total 1200 s.

Query, delay, budget, or Region-related reasons can trigger another attempt after 600 s, allowing transient conditions time to settle. The default limit is three attempts, with a 7200 s deadline from assessment creation to bound processing; the deadline is not a mandatory wait. These durations are engineering settings, not experimentally optimized values or guarantees of event arrival. Query scope specifies account, Region, interval, event family, and source [8]. Stored metadata excludes secret values and IAM secret-key material.

5.2 Evidence-status semantics

A signal is a matched historical event classified by an investigator. Coverage records query/onset limitations; scope-qualified conclusions apply only to the recorded query scope. KNOWN_TRANSITION denotes evidenced onset; BOUNDED_UNKNOWN denotes onset bounded by a verified prior-safe baseline; UNKNOWN lacks this support. Matching signals yield ACTIVITY_OBSERVED; SUCCESSFUL_USE marks a matching successful-use event. No matching activity and no coverage flags yields NO_ACTIVITY_OBSERVED; unresolved limitations yield INCONCLUSIVE. COMPLETE_WITHIN_SCOPE means no recorded coverage-reason flags, including unknown onset, query limits, errors, or delivery delay. A final assessment has no further automatic attempt scheduled; E4 requires final coverage. BEST_EFFORT_NOT_ATTESTED explicitly separates query completion from any guarantee that all AWS events have arrived.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

5.3 Version-specific audit retention

Delivery uploads versioned JSON, records the submitted-byte SHA-256 and VersionId, and checks version-specific GOVERNANCE retention [10]. E4 separately retrieves stored bytes and checks their digest and linkage. The store is in the same account: retention restricts deletion but permits authorized governance bypass. CloudTrail signed-digest validation is a separate source-log mechanism [9], not an operation established by E4.

5.4 Dashboard, notifications, and access separation

A password-authenticated Cloudflare Worker reads records through a Lambda API; SNS notifies operators. Neither presentation component authorizes mutation or serves as ground truth. Decisions, finalized evidence, and verified audit receipts remain distinguishable.

6 Experimental Methodology

6.1 Environment and test fixtures

A fixture is a disposable test resource; a runner is the script that creates test conditions, invokes checks, and records outcomes. All experiments used exactly scoped, tagged fixtures in one AWS account and us-east-1: an IAM user, role and test principals, and dummy secret. E1/E3 used normal paths with hooks disabled; E2 used bounded hooks and authorization failures.

E1 evaluates 90 remediation trials. E2 evaluates 24 safety/failure scenarios: 18 live scenarios and six synthetic exclusions. E3 evaluates nine re-violation pairs comprising 18 correction episodes. E4 evaluates 18 exposure windows (nine positive, nine negative) and 36 audit versions. Offline controls are reported separately. E4 uses the isolated exposure protocol in Section 6.5.

6.2 E1: remediation and timing

An E1 trial introduced one supported violation, awaited the episode decision, checked AWS state through the runner, and restored the fixture. Independent verification here means a separate caller-side check, sharing AWS as the underlying source. A pass required FIXED, engine-attributed resolution, VERIFIED write state (postcondition read-back passed), a verification receipt, and caller-side compliance.

L_episode is the Lambda-generated episode.updated_at minus caller-recorded violation-API completion. It measures recorded completion and includes an unquantified cross-clock offset. L_poll uses caller-side timestamps for the first safe observation minus violation completion and includes five-second polling. These distinguish system-recorded completion from externally observed recovery. Quartiles use linear interpolation at $(n-1)q$; IQR is $Q3-Q1$. Median 95% intervals use 10,000 bootstrap resamples, seed 1234. Values are rounded to 0.1 s. Exact binomial intervals assume comparable independent trials; shared fixtures limit that interpretation.

6.3 E2: no-write decisions, failure injection, and positive controls

E2 ran three cases in each of eight categories (Table 4). Six synthetic nonexistent-target cases exercise pre-read exclusions; 18 use live fixtures. PASS requires the expected decision and applicable resource outcome. A denied write cannot pass as FIXED.

TOCTOU cases use the fixture synchronization hook. Permission-denial cases temporarily apply exact-target IAM Deny policies, check authorization, and restore prior policy. Concurrency cases invoke one target twice while ownership is held, testing exclusion by an existing lease rather than simultaneous claim acquisition. Cleanup is recorded.

6.4 E3: corrected-state persistence and re-violation

An E3 pair comprises initial correction and correction of a reintroduced violation on the same resource. Delays of 10, 60, and 300 s exercise recurrence at different intervals, without estimating an optimal delay. Each pair required distinct episode IDs and caller-side compliant reads. Three offline engine controls test persistently violating postreads, pre-write state change, and valid postreads; they are not additional live pairs.

6.5 E4: exposure investigation and audit checks

E4 used three positive and three negative windows per playbook. A seed is a deliberately executed test action: STS GetCallerIdentity with a disposable key, STS AssumeRole by an unapproved role, or GetSecretValue on a dummy secret [16]. Ground truth here is the runner's successful request correlated by request ID with a CloudTrail event, not an independent telemetry source. A positive pass required account/target/identity/time agreement and that exact event ID in a persisted SUCCESSFUL_USE signal. Negatives omitted matching seeds and required final NO_ACTIVITY_OBSERVED with

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

COMPLETE_WITHIN_SCOPE. No approved-principal, wrong-target, failed-call, or out-of-window decoys were deliberately seeded.

E4 disabled one control-specific rule at a time, then waited 90 s for already-triggered remediation to drain before introducing exposure. Role/secret positives waited 120 s after temporary permission grants to allow authorization propagation before seeding. These are fixture-isolation settings, not measured delivery bounds. The runner directly dispatched remediation and retrieved exact S3 versions to check SHA-256, linkage, and retention. Rule/fixture restoration and temporary-key deletion were checked afterward. This protocol evaluates controlled investigation, not normal event-driven latency.

7 Results

7.1 E1: observed remediation outcomes

All 90 trials passed with FIXED episodes and caller-side compliant AWS states, 30/30 per control. Episode medians were approximately 3 s; polling-observed medians were 7–8 s (Tables 2–3). Five-second polling affects this gap, which does not directly measure additional AWS mutation delay.

Table 2. E1 episode-completion latency from violating API completion (seconds).

Playbook	n/N	Min	Q1	Med.	Q3	Max	IQR
IAM access key	30/30	2.4	3.0	3.2	3.6	5.1	0.6
IAM role trust	30/30	2.4	2.6	2.9	3.3	4.8	0.6
Secrets Manager policy	30/30	2.2	2.7	2.9	3.3	3.7	0.6

Table 3. E1 polling-observed safe-state latency (seconds).

Playbook	Min	Q1	Med.	Q3	Max	Median 95% CI
IAM access key	6.4	6.5	7.2	7.2	8.0	6.5–7.2
IAM role trust	6.4	7.2	8.2	8.3	9.5	7.4–8.3
Secrets Manager policy	6.4	7.1	8.0	8.2	8.8	7.2–8.2

Median per-trial L_poll−L_episode was 3.8 s for keys and 4.8 s for role/secret policies; these differences are not exposure estimates. Exact two-sided 95% pass-rate intervals are approximately 88.4–100% per control and 96.0–100% overall, conditional on independence assumptions.

Initial CloudTrail linkage was 28/30 for keys and 30/30 for each other control. The two unlinked key trials still passed correction checks. Their linkage shortfall has no established root cause in the recorded results and cannot be attributed specifically to delivery lag or permanent event loss.

7.2 E2: safety-gate and failure-control outcomes

All 24 E2 cases met their expected outcomes and cleanup checks (Table 4). The heterogeneous total is descriptive, not a pooled security-accuracy estimate.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

Table 4. Completed E2 categories and bounded interpretation.

Category	Passed / run	Property exercised
Outside-scope exclusion	3/3	No unauthorized target action.
Never-touch protection	3/3	Protected identity not modified.
Unapproved opt-out	3/3	Review/escalation instead of silent exemption.
Secret deletion authorization	3/3	Deletion requires explicit approval.
Pre-mutation state change	3/3	Controlled TOCTOU change blocks stale write.
Concurrent ownership	3/3	Single episode ownership and safe decision behavior.
Authorized positive control	3/3	Eligible scoped violation can be corrected.
IAM permission denial	3/3	Denied write is not misreported as FIXED.
TOTAL	24/24	Controlled evaluation; not broad policy completeness.

Synthetic exclusions establish the exercised pre-read logic; live cases establish fixture-specific outcomes. Neither is evidence of unrestricted authorization correctness.

7.3 E3: re-violation and renewed correction

All nine E3 pairs passed: three per control at 10, 60, and 300 s. Each produced two distinct correction episodes, giving 18 verified outcomes (Table 5). All three offline controls also passed; fixture restoration was recorded.

Table 5. E3 paired re-violation evaluation.

Playbook	Delay values (s)	Pairs passed	Expected correction episodes
IAM access key	10, 60, 300	3/3	6
IAM role trust	10, 60, 300	3/3	6
Secrets Manager policy	10, 60, 300	3/3	6
TOTAL	—	9/9	18

A previous FIXED outcome therefore did not prevent correction of a subsequent violation in the nine tested pairs. Separate caller-side AWS reads confirmed the resulting states; both engine and runner nevertheless relied on AWS API responses.

7.4 E4: historical activity and audit integrity

E4 completed 18/18 windows. All nine positive seed events were confirmed by request-correlated CloudTrail ground truth and matched exactly in persisted investigator signals. All nine negatives yielded NO_ACTIVITY_OBSERVED. Every assessment had COMPLETE_WITHIN_SCOPE coverage, empty coverage reasons, and a known transition-based start (Table 6).

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

Table 6. E4 seeded activity, scoped negative controls, and independently checked audit versions.

Playbook	Positive	Negative	Event-ID matched	Audit versions
IAM access key	3/3	3/3	3/3	12/12
IAM role trust	3/3	3/3	3/3	12/12
Secrets Manager policy	3/3	3/3	3/3	12/12
TOTAL	9/9	9/9	9/9	36/36

All windows recorded CSPM_VERIFIED, the runner classification for verified engine resolution, with no unresolved or external closures. Both audit record kinds passed retrieval, digest, linkage, and active-retention checks: 36 distinct versions. Final records show three rules enabled, three fixtures safe, six keys deleted, and temporary seed permissions removed.

8 Discussion and Limitations

8.1 Supported implementation properties

E1 demonstrates fixture correction with caller-side verification; E2 exercises selected exclusions, state-change handling, held-lease protection, and denied writes. E3 demonstrates renewed correction and includes an offline persistently-violating-postread control. E4 demonstrates exact matching of nine seeded historical events and checks both audit record kinds across 18 windows. Scheduled scanning, stalled-evidence recovery, and interrupted-mutation reconciliation are implemented but lack dedicated end-to-end experiments in E1–E4.

8.2 Measurement and sampling limits

Results concern one account, Region, fixture cohort, policy family, and low-contention workload. No clock-offset measurement accompanies L_episode; neither timing metric identifies the exact enforcement instant. Repeated trials share infrastructure and credentials. E4 has three positives and three negatives per control; all-pass counts describe these cases rather than general security or detection accuracy.

8.3 Security and correctness boundaries

Final reads do not close every TOCTOU interval. Instrumented state-change tests and offline controls do not establish simultaneous claim-race correctness. An ablation would disable one mechanism to measure its contribution; no revalidation-disabled ablation was run. Live workload dependencies and broader policy forms are untested, so disposable-key results do not establish safe deactivation of operational credentials.

8.4 Evidence and retention boundaries

E4 establishes recognition of configured benign API forms. GetCallerIdentity establishes key use, not access to protected data. Investigator and seed-event lookup both depend on CloudTrail; matching does not establish telemetry completeness. Empty negative windows offer limited discrimination evidence without deliberate decoys. No delayed-event, missing-telemetry, or truncated-query injection validates the wait and coverage logic. S3 checks concern inspected versions at check time, with no tamper-injection test or protection against authorized governance bypass.

8.5 Comparative evaluation and reproducibility

No matched baseline quantifies relative performance or safety. Independent replication also remains unestablished: it requires the source revision, Terraform inputs, experiment runners, and raw records, together with an isolated AWS account. Broader accounts, policy forms, workloads, and fault conditions are needed to assess transfer beyond these fixtures.

9 CONCLUSION

The prototype links postcondition-verified AWS remediation episodes to bounded historical investigation and version-specific audit delivery. Experiments recorded 90/90 E1 trials, 24/24 E2 scenarios, nine successful E3 pairs, and 18/18 E4 windows with nine exact positive event matches, nine scoped negatives, and 36 checked audit versions. These results demonstrate the integrated

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

workflow on controlled fixtures. Delayed telemetry, workload dependencies, recovery under faults, and simultaneous claim races remain priorities for further evaluation.

REFERENCES

[1] Amazon Web Services: Setting Up Auto Remediation for AWS Config. AWS Config Developer Guide. https://docs.aws.amazon.com/config/latest/developerguide/setup-autoremediation.html (accessed 21 October 2026).

[2] Cloud Custodian: Cloud Custodian Documentation. https://www.cloudcustodian.io/docs/index.html (accessed 21 October 2026).

[3] Amazon Web Services: Automated Security Response on AWS, Solution Overview. https://docs.aws.amazon.com/solutions/latest/automated-security-response-on-aws/solution-overview.html (accessed 21 October 2026).

[4] Prowler: Find, Prioritize and Fix Cloud Security Risks. https://prowler.com/ (accessed 21 October 2026).

[5] van Ede, T., Khasuntsev, N., Steen, B., Continella, A.: Detecting Anomalous Misconfigurations in AWS Identity and Access Management Policies. In: CCSW 2022, pp. 63–74. ACM (2022). https://doi.org/10.1145/3560810.3564264.

[6] Cook, B.: Formal Reasoning About the Security of Amazon Web Services. In: Chockler, H., Weissenbacher, G. (eds.) CAV 2018, LNCS, vol. 10981, pp. 38–47. Springer (2018). https://doi.org/10.1007/978-3-319-96145-3_3.

[7] Amazon Web Services: Understand How IAM Access Analyzer Findings Work. https://docs.aws.amazon.com/IAM/latest/UserGuide/access-analyzer-concepts.html (accessed 21 October 2026).

[8] Amazon Web Services: Working with CloudTrail Event History. https://docs.aws.amazon.com/awscloudtrail/latest/userguide/view-cloudtrail-events.html (accessed 21 October 2026).

[9] Amazon Web Services: Validating CloudTrail Log File Integrity. https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-log-file-validation-intro.html (accessed 21 October 2026).

[10] Amazon Web Services: Object Lock Considerations. Amazon S3 User Guide. https://docs.aws.amazon.com/AmazonS3/latest/userguide/object-lock-managing.html (accessed 21 October 2026).

[11] Daubner, L., Matulevičius, R., Buhnova, B., Antol, M., Růžička, M., Pitner, T.: A Case Study on the Impact of Forensic-Ready Information Systems on the Security Posture. In: Advanced Information Systems Engineering, CAiSE 2023. LNCS, vol. 13901, pp. 522–538. Springer (2023). https://doi.org/10.1007/978-3-031-34560-9_31.

[12] Alharthi, D., Kawaminami Garcia, I.R.: Cloud Investigation Automation Framework (CIAF): An AI-Driven Approach to Cloud Forensics. arXiv preprint arXiv:2510.00452 (2025). https://doi.org/10.48550/arXiv.2510.00452.

[13] Srinivasan, A., Paramadilok, J.: CUSTOS: Toward Forensic-Ready Zero Trust at the Capture–Containment Boundary. arXiv preprint arXiv:2608.17068v3 (2026). https://arxiv.org/abs/2608.17068v3.

[14] Agade, A.D., Balpande, S.: From Incident to Evidence: Autonomous AIOps for DORA-Ready Financial DevOps Self-Healing Remediation with Traceability and Audit-Grade Proof. International Journal for Multidisciplinary Research 7(6), IJFMR250672540 (2025). https://doi.org/10.36948/ijfmr.2025.v07i06.72540.

[15] Amazon Web Services: DynamoDB Transaction APIs. https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/transaction-apis.html (accessed 21 October 2026).

[16] Amazon Web Services: Log AWS Secrets Manager Events with AWS CloudTrail. https://docs.aws.amazon.com/secretsmanager/latest/userguide/monitoring-cloudtrail.html (accessed 21 October 2026).

(This work is licensed under a Creative Commons Attribution 4.0 International License.)