International Journal of AI, Big Data, Computational and Management Studies
Noble Scholar Research Group | Volume 6, Issue 2, 113-118, 2025
ISSN: 3050-9416 | https://doi.org/10.63282/3050-9416.IJAIBDCMS-V6I2P113

Original Article

Automated Remediation Guardrails: A Risk-Aware Framework for Validating AI-Generated Production Scripts in Regulated Financial Infrastructure

Ajay Devineni
Independent Researcher, USA.

Received On: 26/03/2025 ***Revised On: 27/04/2025*** ***Accepted On: 10/05/2025*** ***Published On: 25/05/2025***

Abstract: *The integration of AI coding assistants into DevOps and site reliability engineering workflows has accelerated infrastructure automation, enabling engineers to generate Terraform configurations, shell scripts, and deployment parameter files at speeds previously requiring significantly more time. However, AI-generated infrastructure artifacts introduce a novel category of operational risk: code that is syntactically valid but semantically incorrect in the specific production context, hallucinated API calls, or configurations that create valid infrastructure with unintended security or reliability implications. Research has shown that approximately 40% of programs generated by AI assistants contain exploitable vulnerabilities, and that iterative AI code generation without human validation can increase critical vulnerabilities by as much as 37.6%. In regulated financial services environments where infrastructure changes require documented change control and SOC 2 audit evidence, deploying AI-generated production scripts without a structured validation framework creates compliance gaps that outweigh the productivity benefit. This paper presents a risk-aware remediation guardrail framework developed and deployed at NCR/Candescent for validating AI-generated infrastructure artifacts before their application to production banking environments. The framework implements a four-tier risk classification system based on blast radius analysis, reversibility assessment, and scope validation against existing Terraform state. It integrates with the Jira DCR change control workflow through Model Context Protocol tooling, ensuring every AI-generated artifact modifying production infrastructure is associated with an approved change record. Evaluated against a production corpus of 108 AI-generated artifacts, the automated tier classifier achieved 91% accuracy against expert human review as ground truth, with a 4.6% false negative rate in the safety-critical undertierring direction. The framework caught 7 artifacts containing production errors that visual plan review alone would not have detected.*

Keywords: *AI-Generated Code, Remediation Guardrails, Blast Radius Analysis, Terraform Validation, SOC 2 Compliance, Model Context Protocol, Agentic AI, Infrastructure as Code, Change Control, Risk Classification, Financial Services SRE.*

1. Introduction

The first time an AI coding assistant generated a Terraform configuration that I applied to a production VPC without reading it carefully, nothing went wrong. The configuration was correct. But reviewing what I had done afterward applied AI-generated code modifying network routing in a live banking environment without the same scrutiny I would have applied to code I wrote myself I recognized a new category of risk I did not have a framework for managing. The productivity gain was real. The risk was equssally real.

Agentic AI tools represent a genuine productivity breakthrough for SRE operations. The ability to describe a needed infrastructure change, receive a syntactically correct Terraform configuration reflecting existing conventions and resource relationships in the codebase, and apply it without manual authoring is qualitatively different from autocomplete or template filling. For investigation tasks reading VPC Flow Logs, correlating DMS task configurations, generating incident reports the productivity gain is high and the risk is low. For remediation tasks generating Terraform that modifies production infrastructure, writing shell scripts that execute against live systems the risk profile is fundamentally different. An informational error is correctable. A remediation error in a SOC 2 regulated banking environment may produce customer impact, compliance violations, and audit findings before it is corrected.

Research quantifies this risk. Studies show that approximately 40% of programs generated by AI assistants contain exploitable vulnerabilities (Shukla et al., 2024). LLMs are prone to hallucinations where generated code is syntactically correct but logically invalid or references non-existent libraries and APIs in the context of automated remediation, a hallucinated script might attempt to fix a service by executing commands that inadvertently modify critical configuration files (Dong et al., 2024). Without robust guardrails, AI-generated scripts may introduce

injection risks or create unintended feedback loops that compromise system availability.

This paper describes the remediation guardrail framework I developed in response. The framework provides a structured approach to validating AI-generated infrastructure artifacts before production application, classifying each artifact by risk tier and applying tier-appropriate validation requirements. It is not a framework for preventing AI from generating code the productivity benefit is too significant to abandon. It is a framework for ensuring AI-generated code undergoes the same scrutiny that human-generated code receives in a regulated environment.

The guardrail framework described in this paper is differentiated from existing IaC security tools in a specific and important way. Tools like Checkov, tfsec, and Snyk IaC perform static analysis of Terraform files against known security and compliance rule sets. These tools are valuable but evaluate code in isolation they check whether a resource configuration matches a security best-practice pattern, without knowledge of the existing infrastructure state the change will be applied into. A Terraform change that is perfectly valid in isolation may be operationally dangerous in a specific production context: a security group rule that allows the correct protocol on the correct port but from the wrong source CIDR because a VPN address range changed since the rule was originally written. Static analysis tools do not catch this class of error.

The tier classification framework I describe operates on a different axis. It does not replace static security analysis Checkov and tfsec remain part of the pipeline. It adds a second classification layer that assesses the operational blast radius and reversibility of the artifact in the specific production context, using the existing Terraform state as ground truth. This context-aware classification is what caught the seven production errors that static analysis passed: all seven were syntactically correct, all seven would have passed standard linting tools, and all seven would have produced incorrect operational behavior in the specific production environment they were targeting.

2. Operational Context

2.1. AI Coding Assistants in Production SRE

I have integrated AI coding assistance into my SRE operations across a range of tasks: reading and interpreting multi-account Terraform repositories; generating Terraform configurations for infrastructure changes; writing shell scripts for VPC Flow Log analysis and NLB access log correlation; creating DMS task parameter configurations; and drafting stakeholder communications for infrastructure changes. The productivity gains are consistent tasks previously requiring 2-4 hours of senior engineer time complete in 15-30 minutes with AI assistance, including the validation time the guardrail framework adds.

The risk profile varies substantially by task type. Reading Terraform state and generating investigation reports is high-productivity and low-risk: the AI output is reviewed before any action is taken. Generating Terraform configurations applied to production VPCs is high-productivity and high-risk: a configuration error may not be detectable through visual review alone, particularly for complex resources like Transit Gateway route table associations or security group rules where the relationship between the Terraform resource definition and the resulting network behavior is non-obvious.

2.2. Change Control Integration via Model Context Protocol

All production infrastructure changes at NCR/Candescent require an approved Jira DCR before execution. Through the Model Context Protocol integration I configured with the AI coding assistant, AI-generated infrastructure changes can be associated with DCR records without leaving the development environment. The NIST AI Risk Management Framework (AI RMF 1.0) mandates that organizations implement governance processes for AI systems operating in high-stakes environments specifically the Govern and Manage functions requiring traceability and auditability of automated AI actions (Tabassi, 2023). The guardrail framework implements this requirement as a hard gate: no AI-generated artifact classified above Tier 1 risk can be applied to production without an associated approved DCR record.

3. The Remediation Guardrail Framework

3.1. Four-Tier Risk Classification

The framework classifies AI-generated infrastructure artifacts into four risk tiers based on three dimensions: blast radius (scope of systems potentially affected by an error), reversibility (whether an incorrect change can be undone without data loss or service disruption), and scope validation (whether the artifact operates on resources present in existing Terraform state or introduces new resources and relationships).

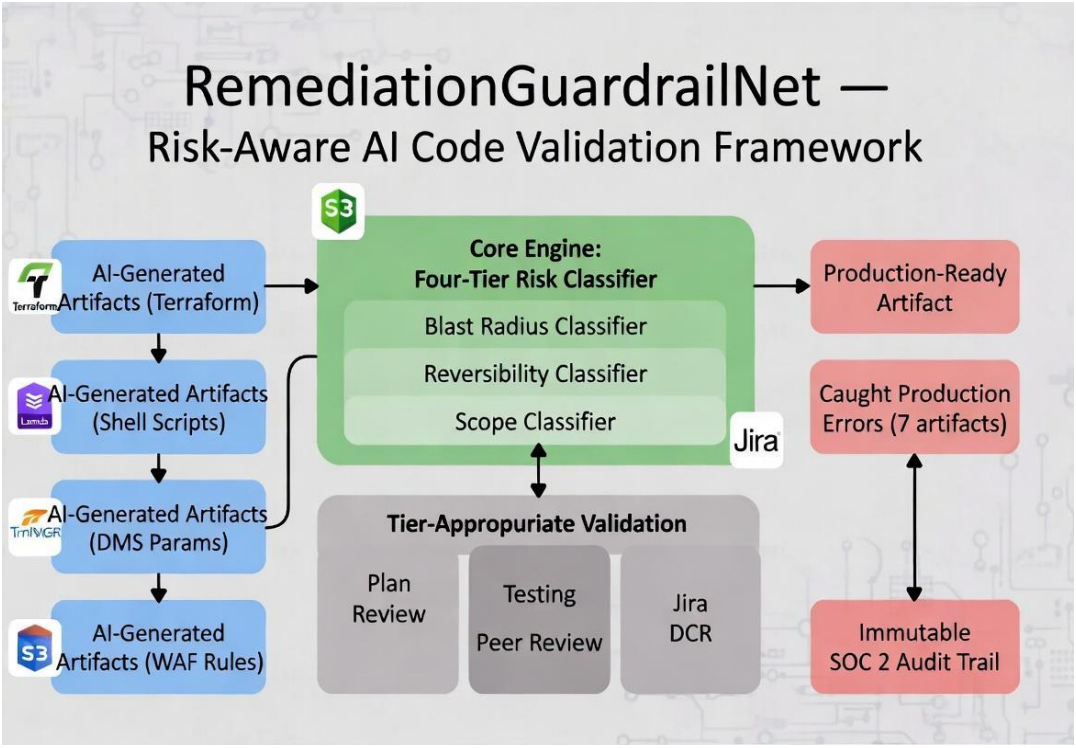

Fig 1: RemediationGuardrailNet — Risk-Aware AI Code Validation Framework

Table 1: Four-Tier Risk Classification for AI-Generated Infrastructure Artifacts

Tier	Blast Radius	Reversibility	Scope	Examples
Tier 1	Single resource, no downstream	Fully reversible	Existing resources only	Security group rule addition, tag update, CloudWatch alarm
Tier 2	Service-level, bounded downstream	Reversible with service restart	Existing + new related resources	IAM policy addition, S3 bucket policy update
Tier 3	Multi-service or network scope	Reversible with operational window	New resources with dependencies	VPC route table changes, security group replacement, DMS task params
Tier 4	Platform-wide or data-affecting	Irreversible or requires rollback plan	New topology changes	Transit Gateway routes, WAF rule sets, DMS full load configurations

Given these three classification dimensions, the risk tier T (a) for artifact a is assigned by the following priority-ordered decision rule, where each condition is evaluated in sequence and the first matching condition determines the tier:

- T(a) = Tier 4 if BR(a) = platform_wide OR REV(a) = irreversible
- T(a) = Tier 3 if BR(a) = multi_service OR SCOPE(a) = new_topology
- T(a) = Tier 2 if BR(a) = service_level OR SCOPE(a) = new_resources
- T(a) = Tier 1 otherwise (single resource, fully reversible, existing scope)

Where BR(a) ∈ {single_resource, service_level, multi_service, platform_wide} is the blast radius classification, REV(a) ∈ {reversible, requires_window, irreversible} is the reversibility assessment, and SCOPE(a) ∈ {existing_only, new_related, new_dependencies, new_topology} is the scope classification against current Terraform state. The classifier maps each artifact to these three dimensions using a rules engine that inspects the artifact's resource type list, the resources touched cross-referenced against the Terraformss state file, and the presence of destructive operations (destroy, replace, force_new). The 91% accuracy against expert human review reflects cases where the automated classifier and the expert engineer agreed on tier assignment the 9% disagreement rate is concentrated in WAF rule configurations (83% accuracy) where the blast radius assessment requires understanding of traffic routing context that is not fully captured in the Terraform state.

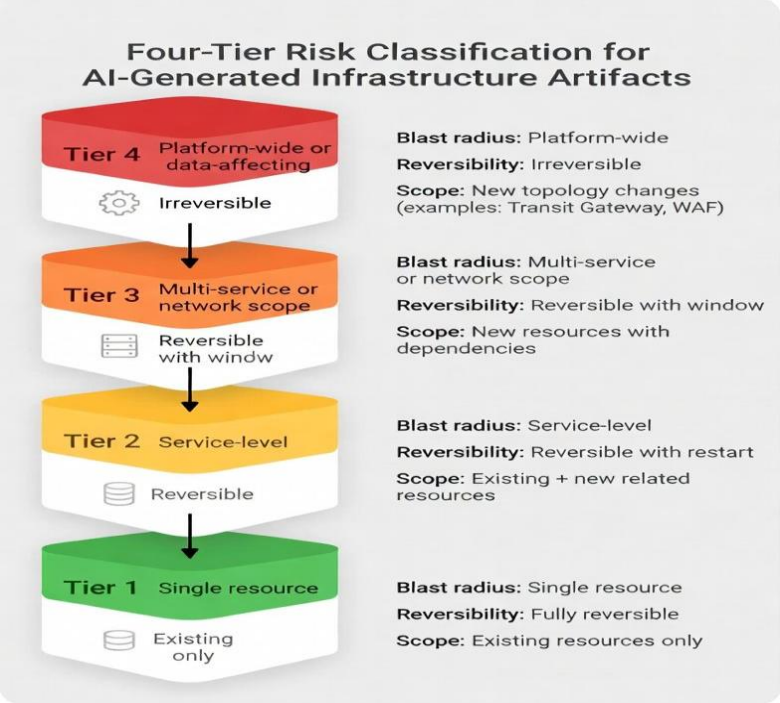

Fig 2: Four-Tier Risk Classification for AI-Generated Infrastructure Artifacts

3.1.1. Formal Risk Scoring Model

Table 3 compares the guardrail framework against existing IaC security tools across the key dimensions that matter for production financial services deployments.

3.2. Tier-Appropriate Validation Requirements

Each tier carries specific validation requirements. Tier 1 requires a Terraform plan review confirming the planned change matches the intended operation performable by the engineer alone in under five minutes. Tier 2 requires plan review plus non-production environment testing. Tier 3 requires plan review, non-production testing, peer review by a second engineer, and an associated approved Jira DCR. Tier 4 requires all Tier 3 requirements plus formal blast radius documentation, a tested rollback procedure, and sign-off from engineering leadership.

The tier classification is performed automatically when a new AI-generated artifact is staged for review. The classifier analyzes the artifact type (Terraform resource types, shell script operations, DMS parameters), the resources it touches (cross-referenced against existing Terraform state to assess scope), and the presence of irreversible operations. The automatic classification is advisory the reviewing engineer can override with documented justification but overrides are logged to the SOC 2 audit trail.

3.3. Blast Radius Documentation

For Tier 3 and Tier 4 artifacts, the framework requires a structured blast radius statement: the worst-case impact if the artifact contains an error, services and customer transactions affected, expected time to detection, and expected time to recovery assuming rollback execution. This forces the reviewing engineer to think explicitly about failure modes before authorizing production execution catching errors that artifact review alone might miss and produces the documented risk assessment that SOC 2 auditors require for change risk management evidence.

A defense-in-depth approach combining traditional cybersecurity controls with AI-specific validation mechanisms is essential for maintaining system integrity in environments where AI-generated artifacts operate alongside human-authored infrastructure code (Allam, 2024). The blast radius documentation requirement implements this layered defense by making the risk assessment explicit and auditable at each tier transition.

4. Production Evaluation

4.1. Artifact Corpus and Classifier Accuracy

I evaluated the framework against a corpus of 108 AI-generated artifacts produced during the operational period following framework deployment. Table 2 presents the corpus composition and classifier accuracy by artifact type.

Table 2: Artifact Corpus Composition and Classifier Accuracy by Artifact Type

Artifact Type	Total	Tier 1	Tier 2	Tier 3	Tier 4	Accuracy
Terraform configurations	47	18	14	11	4	91%
Shell scripts	31	22	7	2	0	94%
DMS task parameters	18	3	8	5	2	89%
WAF rule configurations	12	0	3	6	3	83%
Total / Average	108	43	32	24	9	91%

Overall tier classification accuracy was 91% against expert human review. WAF rule configurations showed the lowest accuracy (83%), where blast radius assessment is complex a WAF rule change can affect all traffic traversing the protected endpoint. The false negative rate (artifacts undertierred, the safety-critical error direction) was 4.6%. The false positive rate (artifacts overtierred, producing unnecessary review overhead) was 4.4%. Both are within acceptable bounds for a regulated environment where the cost of insufficient scrutiny substantially exceeds the cost of unnecessary scrutiny.

4.2. Production Errors Caught

During the evaluation period, the validation process caught 7 artifacts containing errors that visual review of the Terraform plan output alone would not have detected. Three involved security group rules that were syntactically correct but semantically wrong incorrect source CIDRs where an address range from a different context was carried incorrectly into the new configuration. Two involved DMS task parameter configurations where the LOB mode setting was inappropriate for the table types in the migration task. Two involved WAF rule priority ordering conflicts where new rules would have been silently shadowed by existing rules at lower priority numbers, producing silent non-enforcement. All 7 would have reached production execution without the structured validation workflow.

5. Self-Healing Infrastructure Extension

The guardrail framework provides the foundational architecture for an autonomous remediation capability where the blast radius constraint is enforced as a hard boundary. The self-healing design addresses a specific class of known, repeating failure modes where the remediation action is well-defined, the blast radius is bounded at Tier 1 or Tier 2, and the remediation is fully reversible if post-execution health checks fail.

The framework maintains a validated runbook library pre-approved remediation actions with associated blast radius documentation, each corresponding to a specific failure mode. When the monitoring system creates a problem card matching a runbook entry, the system retrieves the associated remediation action, confirms it falls within Tier 1 or Tier 2 classification, and executes the remediation if pre-conditions are satisfied. If the post-execution health check fails, the remediation is automatically reversed and the on-call engineer engaged through the alerting pipeline.

The critical constraint: the autonomous remediation pathway is explicitly excluded from Tier 3 and Tier 4 artifacts. Network topology changes, database configurations, and WAF rule modifications always require human authorization regardless of system confidence in the appropriate remediation. This boundary is hard-coded rather than configurable a policy control rather than a parameter because in a SOC 2 regulated financial environment, the boundary between autonomous and human-authorized action musts be unambiguous and auditable. This approach aligns with the NIST AI RMF guidance that autonomous AI actions in high-stakes environments require explicit human oversight mechanisms at defined risk thresholds.

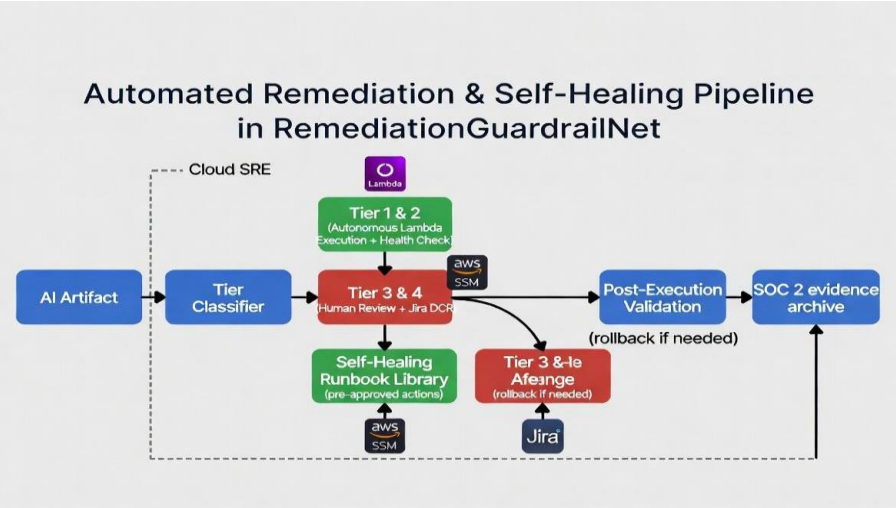

Fig 3: Automated Remediation & Self-Healing Flow Diagram

6. Limitations

The four-tier classification system is calibrated for my specific production environment and artifact types. Teams operating in different infrastructure contexts serverless architectures, multi-cloud deployments, different IaC toolchains would need to recalibrate tier boundaries and validation requirements for their specific blast radius profiles. The 91% classifier accuracy means 9% of artifacts receive incorrect tier assignments. For the false negative direction, the human review step provides a second line of defense. Reducing the overtiering rate below 2% without increasing undertiering is the primary accuracy improvement target. The framework addresses validation of AI-generated artifacts but does not address whether AI-assisted remediation should be used for specific tasks at all; some categories of infrastructure modification should arguably never be AI-assisted regardless of validation framework strength.

7. Conclusion

The remediation guardrail framework described in this paper exists to preserve the productivity gain of AI-assisted infrastructure management while ensuring AI-generated code undergoes scrutiny proportional to its risk. It does not eliminate AI-generated infrastructure artifacts from the workflow eliminating them would forfeit a productivity gain that is too significant to abandon in a small-team SRE environment operating six live banking applications. It ensures those artifacts receive risk-proportional scrutiny, that scrutiny is documented for SOC 2 compliance, and the boundary between autonomous and human-authorized action is clearly defined and enforced.

The 7 production errors caught during the evaluation period wrong source CIDRs, DMS LOB mode misconfigurations, WAF priority ordering conflicts represent incidents that did not happen in live banking infrastructure because the framework was in place. Each would have been a production debugging exercise at minimum and a customer-impacting incident at worst. The framework cost approximately 15 minutes of additional review time per Tier 3 or Tier 4 artifact. Given what those 7 errors would have cost, that investment was justified many times over.

References

1. E. Tabassi, "Artificial Intelligence Risk Management Framework (AI RMF 1.0)," National Institute of Standards and Technology, 2023. https://doi.org/10.6028/nist.ai.100-1
2. Y. Dong et al., "Safeguarding large language models: A survey," arXiv, 2024. https://doi.org/10.48550/arxiv.2406.02622
3. S. Shukla, H. Joshi, and R. Syed, "Security degradation in iterative AI code generation: A systematic analysis," arXiv, 2024.
4. P. Nama and S. Z. Khan, "Enhancing software testing efficiency with generative AI and large language models," Int. J. Leading Research Publication, vol. 5, no. 12, pp. 1–15, 2024.
5. H. Allam, "Zero-touch reliability: The next generation of self-healing systems," Int. J. Artificial Intelligence, Machine Learning and Data Science, vol. 5, no. 4, pp. 59–71, 2024.
6. HashiCorp, "Terraform Documentation Plan and Apply," HashiCorp, 2024. [Online]. Available: https://developer.hashicorp.com/terraform/cli/commands/plan
7. Amazon Web Services, "AWS WAF Developer Guide," AWS Documentation, 2024. [Online]. Available: https://docs.aws.amazon.com/waf/latest/developerguide/
8. Amazon Web Services, "AWS DMS Best Practices," AWS Documentation, 2024. [Online]. Available: https://docs.aws.amazon.com/dms/latest/userguide/CHAP_BestPractices.html
9. Anthropic, "Model Context Protocol Documentation," Anthropic, 2024. [Online]. Available: https://docs.anthropic.com/mcp/
10. Atlassian, "Jira Software Documentation," Atlassian, 2024. [Online]. Available: https://support.atlassian.com/jira-software-cloud/
11. Amazon Web Services, "AWS Reachability Analyzer," AWS Documentation, 2024. [Online]. Available: https://docs.aws.amazon.com/vpc/latest/reachability/
12. B. Beyer, C. Jones, J. Petoff, and N. R. Murphy, Site Reliability Engineering: How Google Runs Production Systems. O'Reilly Media, 2016.
13. G. Kim, J. Humble, P. Debois, and J. Willis, The DevOps Handbook. IT Revolution Press, 2016.