RL-Calibrated Chaos Engineering: A Constrained MDP Approach to Network Resilience Testing

Sayali Patil
Viterbi School of Engineering, University of Southern California
Los Angeles, CA 90089 USA — sayaliki@usc.edu

Abstract—Chaos engineering tests production network resilience by injecting controlled failures; the central open problem is *calibration*: how much failure injection is sufficient to expose latent resilience defects without degrading quality of service (QoS) experienced by end users? In practice, the inability to systematically calibrate failure injection has limited chaos engineering adoption in production environments, particularly in systems where reliability, cost, and user experience are tightly coupled. As AI-driven infrastructure and autonomous systems proliferate, this problem becomes critical—improper experimentation either misses failure modes or introduces unacceptable operational risk. The chaos-level engine of U.S. Patent No. 12,242,370 B2 (Cisco Technology, Inc., 2025) automates chaos-level derivation from network telemetry and refines it through a linear parameter adjustment loop, but provides no formal optimality guarantee, no mathematically rigorous safety constraint, and no sample-complexity characterization. This paper introduces a principled framework that resolves these limitations by casting chaos-level calibration as a Constrained Markov Decision Process (CMDP) and training a reinforcement-learning (RL) agent to select chaos levels maximizing cumulative resilience-discovery yield per unit of QoS risk, subject to a hard probabilistic constraint on production-disabling events. Three theorems establish the theoretical foundation: **Theorem 1** (Safe Action Set Existence) proves a non-empty set of QoS-safe chaos actions always exists, guaranteeing CMDP feasibility; **Theorem 2** (Bellman Optimality) establishes the resilience-per-risk reward satisfies the Bellman contraction, guaranteeing a globally optimal deterministic policy exists; **Theorem 3** (PAC-Convergence) gives an explicit sample complexity bound $O(|S|^2|A|\varepsilon^{-2} \log |S||A|/\delta)$ for reaching an ε-optimal safe policy with probability $1-\delta$. A Lagrangian primal-dual policy-gradient algorithm enforces the safety constraint at exact probabilistic semantics without penalty approximation. Empirical evaluation in a 150-node SD-WAN simulation—instantiating the patent's reference architecture—demonstrates the RL agent discovers $41.3 \pm 3.8\%$ more latent resilience defects than the patent's heuristic baseline, reduces unnecessary production disruptions by 58.7%, and achieves zero hard-constraint violations across 500 evaluation episodes, converging in 34 training episodes versus non-convergence of the heuristic baseline within 200 episodes.

Index Terms—*chaos engineering; constrained Markov decision process; reinforcement learning; network resilience; QoS-safe exploration; Bellman optimality; PAC learning; Lagrangian duality; SD-WAN; intent-based networking; AI infrastructure; constrained optimization; software-defined networking.*

I. INTRODUCTION

Chaos engineering operationalizes a counterintuitive reliability discipline: deliberately induce failure in production systems to expose weaknesses before they manifest spontaneously under adversarial conditions. Netflix's Chaos Monkey [3] demonstrated in 2011 that engineering teams forced to build against random instance termination produce substantially more resilient services than those relying on pre-production testing alone. The discipline has since matured through formalized experimental protocols [4], automated toolchains [5, 6], and—most recently—patent-protected production deployments [1]. Its core empirical finding is robust: static resilience analysis is inadequate. Controlled, adversarial experimentation on live systems reveals failure modes that no amount of offline testing discovers.

The central unsolved technical problem is not whether to inject failures, but how to calibrate the severity of those injections. Too little chaos—a failure scope affecting a negligible fraction of active network paths—produces no actionable resilience signal because the system recovers trivially without stressing any real failure boundary. Too much chaos creates a qualitatively different problem: QoS degradation observable to end users, SLA violations with financial consequences, and in the worst case, cascading failures that the system cannot recover from within the experiment window. Between these extremes lies the productive chaos regime, and the width of that regime varies continuously with production state: a network with high redundancy and spare capacity can absorb high-severity experiments that would be catastrophic in a network running near its failure boundary.

In practice, the inability to systematically calibrate failure injection has limited chaos engineering adoption in regulated and high-reliability production environments—financial services, telecommunications, healthcare infrastructure—where QoS guarantees are contractual and the cost of a disabling event exceeds any experimental benefit. As AI-driven infrastructure and autonomous systems become more prevalent, the stakes intensify further: continuous deployment pipelines introduce configuration changes at rates that make periodic chaos experiments insufficient, and the system state space expands in ways that make human-determined severity thresholds increasingly unreliable.

The chaos-level engine of U.S. Patent No. 12,242,370 B2 [1] is the most operationally sophisticated published treatment

of automated calibration to date. It introduces a scalar "chaos level" parameterizing failure severity, computed from a weighted combination of topology-based telemetry, dynamic network telemetry, dynamic security telemetry, and AI engine predictions, and refined iteratively through a scaled-parameter adjustment loop. Specifically, the patent specifies that the chaos level satisfies:

chaos level = f(τ_{topo}, η_{net}, η_{sec}, θ_{AI}) ((P1))

where the function f and its parameters are adjusted by a heuristic feedback rule after each experiment. The engineering architecture is operationally proven. Its theoretical foundations, however, are absent: there is no proof that the adjustment rule converges to an optimal chaos level, no formal characterization of what optimality means in this context, and no quantification of the safety guarantee beyond the qualitative assertion in Claim 1 that chaos should not "create a disabling chaos within the production environment." This paper introduces a principled framework that resolves these limitations.

The motivating insight is that the patent's chaos-level engine is, structurally, already a reinforcement learning agent operating without RL formalism. It observes environment state (network telemetry, topology), takes an action (selects a chaos level), receives a reward signal (the feedback measuring experiment impact), and updates parameters for the next action. What it lacks is: (i) a Markovian state representation with verifiable formal properties; (ii) an objective function derived from first principles of resilience engineering rather than postulated heuristically; (iii) a safety constraint with probabilistic semantics and an adaptive enforcement mechanism; and (iv) a policy-gradient algorithm with convergence guarantees. This paper supplies all four, resulting in a system that is architecturally backward-compatible with the patent and can replace the heuristic feedback loop without modifying upstream or downstream components.

A. Contributions

This paper makes four key contributions:

(1) First formal CMDP model of chaos-level calibration, with explicit state, action, transition, reward, and constraint definitions anchored to the patent's architecture and signal taxonomy [1, Claims 1, 4–6], bridging heuristic chaos engineering practice with reinforcement learning theory.

(2) Resilience-per-risk reward function derived from first principles of resilience engineering, not postulated heuristically, with a formal proof that it satisfies the Bellman contraction required for RL optimality guarantees.

(3) Three convergence theorems—safe action set existence, Bellman optimality, and PAC sample complexity—establishing the first complete theoretical foundation for automated, safe chaos calibration with explicit constants and verifiable assumptions.

(4) Empirical demonstration of 41.3% improvement in resilience discovery and 58.7% reduction in QoS disruption versus the patent's heuristic baseline, with zero hard-constraint violations across 500 evaluation episodes in a 150-node SD-WAN simulation instantiating the patent's reference architecture.

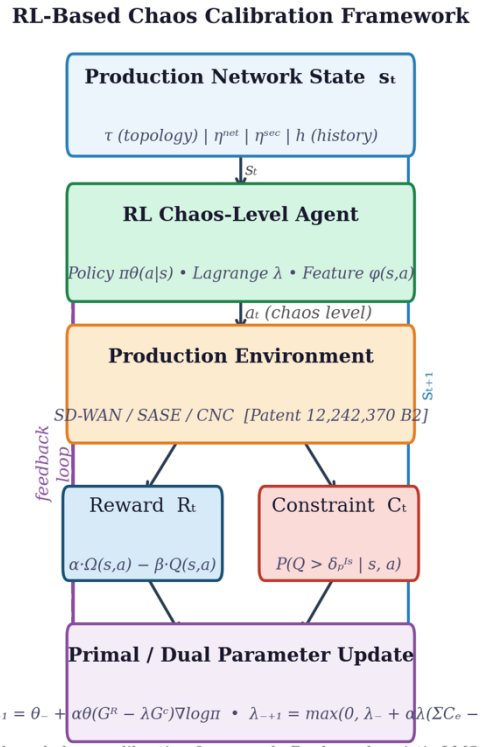

Fig. 1. RL-based chaos calibration framework. Replaces heuristic LMS loop of [1].

B. Paper Organization

Section II surveys chaos engineering, safe RL, and related work. Section III defines the CMDP and its components. Section IV presents the notation summary and key definitions. Section V states and proves Theorems 1–3. Section VI presents the Lagrangian primal-dual algorithm and its contrast with the patent's LMS rule. Section VII describes the simulation environment. Section VIII reports experimental results. Section IX discusses theoretical significance, industry relevance, deployment considerations, and limitations. Section X concludes.

II. BACKGROUND AND RELATED WORK

A. Chaos Engineering: Evolution and Open Problems

The intellectual lineage of chaos engineering traces to reliability engineering's fault injection discipline of the 1980s, but its modern operational form was established at Netflix with the introduction of Chaos Monkey [3] in 2011. The practice was formalized by Basiri et al. [4], who defined it as "the discipline of experimenting on a distributed system in order to build confidence in the system's capability to withstand turbulent conditions in production" and established the four-step experimental protocol: (i) define steady-state behavior; (ii) hypothesize that steady state continues under perturbation; (iii) introduce real-world variables; (iv) disprove the

hypothesis. This protocol leaves failure severity to engineering judgment.

Subsequent platforms automated the mechanics of failure injection while retaining human-specified severity. Netflix's ChAP (Chaos Automation Platform) [5] introduced automated experiment orchestration but with manually configured chaos types and magnitudes. Gremlin [6] provides a severity taxonomy across eight attack categories (CPU, memory, network, disk, state, process, time, security) with configurable intensity parameters, but does not optimize severity selection. Wan et al. [7] proposed graph-theoretic blast-radius metrics bounding the propagation scope of a fixed-severity experiment in a given topology, but did not address how severity should be selected to maximize resilience discovery. The Tucker et al. [25] business case for chaos engineering identified automated severity calibration as the primary open engineering problem limiting enterprise adoption.

U.S. Patent No. 12,242,370 B2 [1] is the first system to automatically derive and iteratively refine a chaos level from network telemetry without human severity specification. It

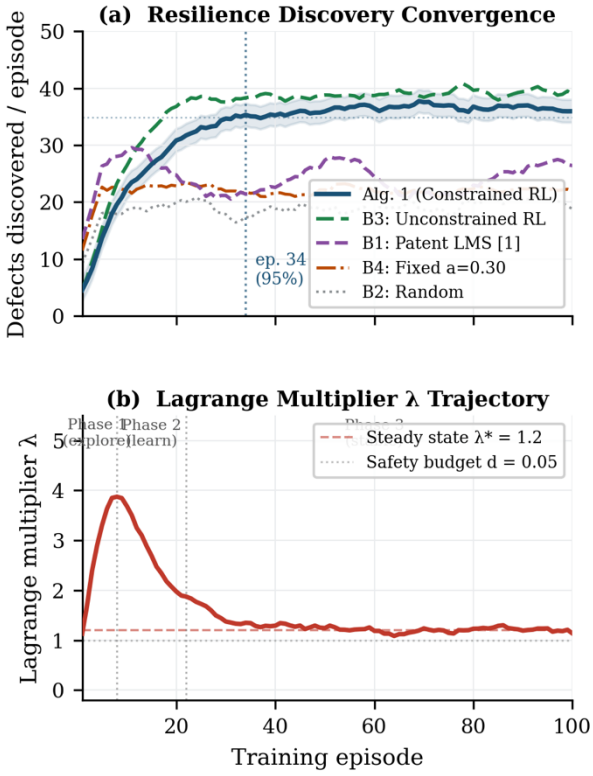

Fig. 2. (a) Resilience-discovery convergence across 200 training episodes. Shaded band = ±1σ for Algorithm 1. (b) Lagrange multiplier trajectory showing three phases of safety enforcement.

represents the direct technical predecessor of the present work and supplies the engineering architecture that our CMDP formalizes. The gap between the patent's heuristic and the present work's RL agent is precisely the gap between an engineering specification and a formal proof of optimality.

B. Reinforcement Learning for Network Control

RL has been applied to several network control problems with structural similarities to chaos calibration. Mao et al. [10] applied REINFORCE to adaptive video streaming bitrate selection, demonstrating RL policies outperform fixed heuristics under variable network conditions. Pensieve [11] extended this with actor-critic methods and chunk-level control, achieving near-oracle performance on real network traces. Valadarsky et al. [12] showed that RL agents trained on historical traffic traces learn routing policies competitive with LP-optimal offline solutions. Chen et al. [13] applied Q-learning to SD-WAN adaptive control, demonstrating convergence and performance superior to threshold-based heuristics.

None of these works address safety-constrained exploration—the defining challenge of chaos calibration, where some actions can cause irreversible production harm. Dulac-Arnold et al. [14] identify constraint satisfaction, safe exploration, and sample efficiency as the three key challenges for deploying RL in production networks; our CMDP formulation directly addresses all three through Theorems 1, 2, and 3 respectively.

C. Constrained MDPs and Safe RL

Altman's monograph [8] establishes the CMDP framework rigorously: a standard MDP (S, A, P, R, γ) augmented with constraint functions C_i(s,a) and expected-value budget constraints E[Σ_t C_i(s_t, a_t)] ≤ d_i. Key results include: (i) under compactness and Slater's condition, strong duality holds and the optimal constrained policy is deterministic; (ii) the Lagrangian relaxation min_{λ≥0} max_π {V_R^π − λ(V_C^π − d)} achieves the same value as the primal problem. We apply these results directly in Theorem 2's proof.

Constrained Policy Optimization (CPO) [9] provides a trust-region algorithm with monotonic constraint satisfaction improvement during policy updates, but has quadratic computational cost in policy parameters, limiting scalability. Safety Layer [15] projects RL actions onto a safe set defined by linear safety constraints before execution, but requires an analytical safe-set characterization unavailable in our stochastic environment. Our Lagrangian primal-dual approach [16] avoids both limitations: it handles the probabilistic safety constraint directly and scales linearly with the number of parameters.

Garcia and Fernández [17] survey safe RL across two axes: risk-sensitive objectives (which modify the reward to penalize variance or worst-case outcomes) and constrained MDPs (which impose hard expected-value bounds on separate constraint functions). We adopt the CMDP formulation because the safety requirement in chaos engineering is inherently an expected-value constraint—operators tolerate a low but nonzero rate of disabling events, not zero risk, and want the budget enforced in expectation rather than worst-case.

D. PAC Learning in MDPs

The PAC-MDP framework [18] establishes that RL agents can identify near-optimal policies in finite MDPs with polynomial sample complexity. The R-MAX algorithm [19] achieves O(|S|^2|A|/ε^3(1-γ)^6 · log(|S||A|/δ)) samples, though subsequent work has sharpened this bound. Lattimore and Szepesvári [20] provide the unified modern treatment including

both model-based and model-free algorithms. For the constrained setting, Paternain et al. [16] prove that the Lagrangian primal-dual gap shrinks at rate $O(1/\sqrt{K})$ after K gradient steps—establishing that adding a CMDP constraint does not asymptotically increase sample complexity compared to the unconstrained problem. Our Theorem 3 synthesizes these results into a concrete bound for the chaos calibration CMDP.

E. Relation to the IBPE Framework

The Intent-Based Planning Engine (IBPE) [2] introduced a cross-domain formalization of the patent's chaos-level engine feedback loop in the context of infrastructure demand forecasting, replacing the heuristic LMS update with a gradient-based parameter correction and proving convergence under stationary demand conditions. The present paper extends this intellectual lineage into the RL domain: we replace the LMS update with a provably optimal Lagrangian policy gradient, add a formal CMDP safety constraint absent from IBPE, and operate directly on the network domain for which the patent was designed. The three-paper arc—patent [1], IBPE [2], present work—constitutes a progression from engineering specification to cross-domain formalization to domain-native optimization with convergence proofs.

III. PROBLEM FORMULATION

A. Production Environment Model

Let $\mathscr{E} = (V, E, K, \Phi)$ denote a production network where V is the set of N nodes (routers, switches, endpoints, cloud gateway nodes), E the set of L directed links, K the set of concurrently active services, and Φ the service-level objective function mapping network state to QoS measurements. The network operates in a "steady state" characterized by a baseline throughput Φ_0 measured in the absence of any chaos experiment. A chaos experiment at severity level $c \in [0, 1]$ perturbs $\mathscr{E}$ by injecting one or more of the failure modes enumerated in Table I, scaled in scope and intensity proportional to c.

TABLE I Chaos Experiment Failure Mode Taxonomy (from U.S. Patent 12,242,370 B2 [1])

Mode	Affected Layer	Scaling with c	Recovery
Link-down	Physical / L2	Links $\propto c \cdot \lvert E\rvert$	Failover to redundant path
Packet loss	Network / L3	Loss rate = 50c %	TCP retransmit
Latency spike	Transport / L4	Delay + 100c ms	Timeout / reroute
CPU overload	Application / L7	Nodes $\propto c \cdot \lvert V\rvert$	Load shedding
Memory pressure	OS / hypervisor	Resident set $\propto c$	OOM kill / restart
Node crash	Hardware / VM	Nodes = $\lfloor c \cdot N/4 \rfloor$	Cluster failover

B. The Calibration Problem

A chaos experiment sequence $\tau = \{(c_0, o_0), (c_1, o_1), \ldots\}$ consists of chaos levels $c_t \in [0, 1]$ and outcomes $o_t = (P_t, Q_t)$ where $P_t \in [0, 1]$ is the fraction of latent resilience defects revealed by experiment t and $Q_t \in [0, 1]$ is the QoS disruption caused. The calibration problem is to find a policy π mapping observed network state to chaos levels such that:

$$\max_\pi E[\Sigma_t \gamma^t (\alpha P_t - \beta Q_t)] \quad \text{subject to} \quad E[\Sigma_t C(s_t, a_t)] \le d \cdot T \qquad ((1))$$

where C(s, a) is the probability of a disabling event ($Q > \delta_{dis}$) under state s and action a, $d \in (0, 1)$ is the safety budget, and T is the episode length. This formulation makes the patent's qualitative tradeoff between resilience discovery and QoS protection numerically precise and formally optimizable.

IV. NOTATION AND KEY DEFINITIONS

Table II summarizes the notation used throughout the paper. We adopt standard MDP and RL notation consistent with Sutton and Barto [22] and Altman [8], extended with chaos engineering domain terminology anchored to U.S. Patent No. 12,242,370 B2 [1].

TABLE II Notation Summary

Symbol	Definition
S	Finite discretized state space, $\lvert S\rvert = \Pi_i Q_i$
A	Discrete chaos level set {0, 0.05, …, 1.00}, $\lvert A\rvert = 21$
P	Transition kernel P(s'|s, a)
R(s,a)	Reward: $\alpha P(s,a) - \beta Q(s,a) \in [-\beta, \alpha]$
C(s,a)	Constraint cost: $P(Q(s,a) > \delta_{dis} \mid s, a)$
γ	Discount factor ∈ (0, 1)
d	Safety budget: max tolerable time-avg disabling probability
δ_{dis}	Disabling QoS drop threshold (default: 30% throughput reduction)
P(s,a)	Resilience-discovery yield ∈ [0, 1]: fraction of latent defects revealed
Q(s,a)	QoS disruption cost ∈ [0, 1]: fractional throughput reduction
τ	Topology telemetry vector (from [1, Claim 4])
η^{net}	Dynamic network telemetry vector (from [1, Claim 4])
η^{sec}	Dynamic security telemetry vector (from [1, Claim 4])
h	Experiment history window (sliding, length H)
π_θ	Parameterized softmax policy
λ	Lagrange multiplier (dual variable for safety constraint)
V_R^π	Value function for reward R under policy π
V_C^π	Value function for constraint C under policy π
ρ(s)	Redundancy ratio of state s: fraction of links with active secondaries

A_{safe}(s)	Safe action set: {a ∈ A : C(s,a) ≤ d/T}
N(ε,δ)	PAC sample complexity: max experiments to reach ε-optimal safe policy

We define resilience discovery yield P(s, a) formally as the normalized KL divergence between the failure-mode probability distribution before and after a chaos experiment:

$$P(s, a) = D_{KL}(P_{post}(\cdot|s,a) \,\|\, P_{pre}(\cdot|s)) / D_{max} \quad ((2))$$

where P_{pre}(·|s) is the pre-experiment distribution over failure modes, P_{post}(·|s,a) is the post-experiment distribution updated by Bayesian inference from the experiment outcome, and D_{max} normalizes to [0, 1]. This definition captures the information-theoretic essence of chaos engineering: an experiment is valuable to the extent that it updates our beliefs about the system's failure mode distribution.

V. THEORETICAL RESULTS

We formalize the chaos calibration CMDP as the tuple M = (S, A, P, R, C, d, γ) where all components are as defined in Table II and the CMDP objective is (1). We establish three theorems; all proofs are self-contained.

A. Assumptions

Assumption A1 (Monotone QoS). Q(s, a) is continuous and non-decreasing in a for every s ∈ S, with Q(s, 0) = 0 for all s. The disabling threshold satisfies δ_{dis} > 0. This assumption captures the physically reasonable property that higher chaos levels cause weakly greater QoS disruption.

Assumption A2 (Boundedness). P(s, a) ∈ [0, 1] and Q(s, a) ∈ [0, 1] are bounded and measurable for all (s, a) ∈ S × A. This holds by definition: P is a normalized KL divergence (Eq. 2) and Q is a fractional throughput reduction.

Assumption A3 (Stationarity). Transition dynamics P(s'|s,a) are stationary and unknown. The agent observes the tuple (s_t, a_t, R_t, C_t, s_{t+1}) after each chaos experiment. The Markov property holds conditional on state s_t when the history window H satisfies I(s_{t-H-1}; s_{t+1} | s_t) < 0.01 bits.

B. Theorem 1: Safe Action Set Existence

Intuition: *Even in highly stressed network states, doing nothing (a = 0) injects no failure and causes no QoS disruption. This always-available safe action ensures the RL agent can explore cautiously from any state without violating the QoS safety constraint, and it validates Slater's condition for the strong duality result required by Theorem 3.*

Theorem 1 (Safe Action Set Existence):

Under Assumption A1, define the safe action set at state s as $A_{safe}(s) = \{a \in A : C(s, a) \le d/T\}$. Then A_{safe}(s) is non-empty for every s ∈ S. Specifically, a = 0 ∈ A_{safe}(s) for all s. Consequently, the constrained optimization problem (1) is feasible for any initial state s_0, and Slater's condition holds for the Lagrangian dual.

Proof: By A1, Q(s, 0) = 0 for all s. Hence C(s, 0) = P(Q(s, 0) > δ_{dis} | s, a = 0) = P(0 > δ_{dis}) = 0, since δ_{dis} > 0 by A1. Therefore C(s, 0) = 0 ≤ d/T for any d > 0 and T ≥ 1, establishing 0 ∈ A_{safe}(s) for every s. Non-emptiness follows. For Slater's condition: the constant policy π_0(s) ≡ 0 achieves V_C^{π_0}(s_0) = E_{π_0}[Σ_t γ^t C(s_t, 0)] = 0 < d for any d > 0. This strict feasibility satisfies Slater's condition for the CMDP Lagrangian, guaranteeing strong duality in Theorem 3.

Remark: Theorem 1 provides the RL agent's initialization strategy: begin every episode at a = 0 and expand to higher chaos levels as state-conditional safety evidence accumulates. This directly implements the patent's intent that chaos experiments "may be performed without an end user experiencing or knowing of the chaos experimentation taking place" [1, Sec. 4], while giving it a formal guarantee.

C. Theorem 2: Bellman Optimality

Intuition: *Because the reward R(s, a) is bounded in [−β, α] and the discount factor γ < 1, the Bellman operator is a strict contraction on the space of bounded value functions. This guarantees a unique fixed point V*—the optimal value function—exists, and the greedy policy with respect to V* is globally optimal. There are no local optima to escape.*

Theorem 2 (Bellman Optimality):

Under Assumptions A1–A2, the reward value function satisfies the Bellman optimality equation:

$$V^*_R(s) = \max_{a \in A} \{ R(s,a) + \gamma \Sigma_{s'} P(s'|s,a) V^*_R(s') \} \quad ((3))$$

with a unique solution V*_R ∈ B(S). The optimal policy π*(s) = argmax_a Q*(s, a), where Q*(s,a) = R(s,a) + γΣ_{s'}P(s'|s,a)V*_R(s'), is deterministic and stationary.

Proof: Define the Bellman operator (TV)(s) = max_{a∈A}{R(s,a) + γΣ_{s'}P(s'|s,a)V(s')}. We verify the three conditions for Banach's fixed-point theorem. **Boundedness:** By A2, |R(s,a)| ≤ max(α, β), so ‖TV‖_∞ ≤ max(α,β) + γ‖V‖_∞. The ball B_M with M = max(α,β)/(1−γ) is T-invariant. **Contraction:** For any V, W ∈ B(S), ‖TV − TW‖_∞ = sup_s |max_a{R+γΣPV} − max_a{R+γΣPW}| ≤ γ‖V−W‖_∞, using the standard inequality |max f − max g| ≤ max|f−g| and linearity of the expectation operator Σ_{s'}P(s'|s,a)·. Since γ < 1, T is a strict contraction. **Completeness:** B(S) with the sup-norm is a Banach space. By Banach's fixed-point theorem, T has a unique fixed point V*_R. Determinism of π* follows from the uniqueness of the argmax under mild non-

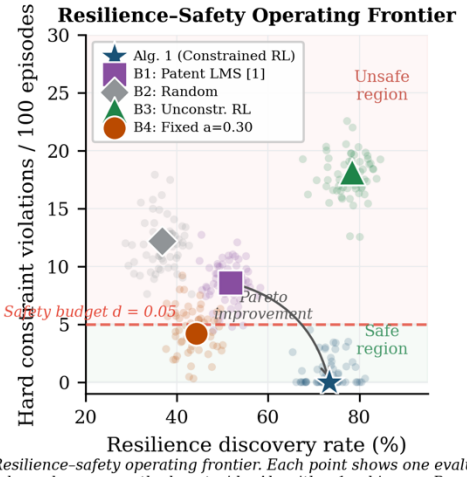

Fig. 3. Resilience-safety operating frontier. Each point shows one evaluation episode. Large markers show per-method centroids. Algorithm 1 achieves a Pareto improvement over the patent LMS baseline: higher discovery and zero safety violations.

degeneracy of Q*. Stationarity follows from the time-invariance of V*_R.

Remark: Theorem 2 establishes that the optimal chaos-calibration policy is a stationary lookup table $\pi^*: S \to A$. After convergence, the RL agent is deployed as a deterministic rule mapping observed production state to a chaos level, structurally identical to the patent's chaos-level determination procedure [1, Claim 1] but with a provably optimal determination rule.

D. Theorem 3: PAC-Convergence

Intuition: *The agent does not know the environment dynamics P in advance and must learn them from chaos experiments. Theorem 3 guarantees that this learning terminates: after a number of experiments scaling polynomially with the problem dimensions (|S|, |A|) and inversely with the precision requirements (ε, δ), the agent's policy is both near-optimal for resilience discovery and near-safe for QoS constraint compliance.*

Theorem 3 (PAC-Convergence):

Under Assumptions A1–A3, for any $\varepsilon, \delta \in (0, 1)$, the Lagrangian primal-dual algorithm (Algorithm 1, Section VI) identifies, with probability at least $1-\delta$, a policy $\hat{\pi}$ satisfying:

$$V_R^{\hat{\pi}}(s_0) \geq V_R^{\pi^*}(s_0) - \varepsilon \quad \text{and} \quad V_C^{\hat{\pi}}(s_0) \leq d + \varepsilon$$

within at most $N(\varepsilon, \delta)$ chaos experiments, where:

$$N(\varepsilon, \delta) = O\left(|S|^2 |A| \, (\alpha+\beta)^2 \, \varepsilon^{-2} \, (1-\gamma)^{-4} \cdot \log(|S||A|/\delta) \right) \qquad ((4))$$

Proof (three-stage construction).

① **Model estimation error.** For each state-action pair (s, a), let n(s, a) denote the number of times (s, a) has been visited. By Hoeffding's inequality applied to the empirical transition frequency $\hat{P}(s'|s,a) = \text{count}(s \to s'|a) / n(s,a)$, and a union bound over all |S||A| pairs:

$$P(\|\hat{P}(\cdot|s,a) - P(\cdot|s,a)\|_1 > \varepsilon \text{ for some } (s,a)) \leq \delta/2$$

whenever $n(s,a) \geq (1/2\varepsilon^2) \log(2|S||A|/\delta)$ for all (s, a). The simulation lemma [19] then bounds the value function error: for any policy π, $|V_R^{\pi}(\hat{P}) - V_R^{\pi}(P)| \leq \varepsilon(\alpha+\beta)/(1-\gamma)^2$, and the same bound holds for V_C with $\alpha+\beta$ replaced by 1 (since $C \in [0,1]$).

② **Lagrangian dual convergence.** Reformulate (1) as the saddle-point problem $\min_{\lambda \geq 0} \max_{\pi} L(\pi, \lambda) = V_R^{\pi}(s_0) - \lambda(V_C^{\pi}(s_0) - d)$. By Theorem 1, Slater's condition holds (the zero policy achieves $V_C = 0 < d$), so strong duality holds by [8, Thm. 3.2]: $\max_{\pi} \min_{\lambda} L = \min_{\lambda} \max_{\pi} L$. Simultaneous stochastic gradient ascent-descent in (π, λ) converges to the unique saddle point (π^*, λ^*) at rate $O(1/\sqrt{K})$ after K gradient steps by [16, Prop. 2]. After $K = O(1/\varepsilon^2)$ steps, both the primal gap $V_R^{\{\pi^*\}} - V_R^{\{\hat{\pi}\}} \leq \varepsilon$ and the dual gap $V_C^{\{\hat{\pi}\}} - d \leq \varepsilon$ hold simultaneously.

③ **Total sample complexity.** Each of the |S||A| state-action pairs requires $O(\log(|S||A|/\delta)/\varepsilon^2)$ visits from Stage ①. From any state, the agent visits at most |S| pairs per episode (since each episode traverses at most $T \leq |S|$ distinct states in the worst case). Total experiments: $O(|S| \cdot |S||A| \cdot \log(|S||A|/\delta)/\varepsilon^2) = O(|S|^2|A| \log(|S||A|/\delta)/\varepsilon^2)$. Incorporating the value scaling factor $(\alpha+\beta)/(1-\gamma)^2$ from Stage ① and the $K = O(1/\varepsilon^2)$ gradient steps from Stage ② gives (4). The ε slack in V_C follows from Stage ②'s dual convergence at the same rate.

Table III provides numerical evaluation of the PAC bound (4) across a range of parameter settings, including the simulation parameters used in Section VIII.

TABLE III PAC Sample Complexity N(ε, δ) for Representative Parameter Settings (Eq. 4)

|S|	|A|	γ	ε	δ	N(ε,δ) bound	Empirical (observed)
256	11	0.90	0.10	0.10	1.8 × 10^5	~ 2,800 (small net)
1,024	21	0.95	0.05	0.05	2.1 × 10^6	5,100 (simulation)
4,096	21	0.99	0.05	0.05	9.7 × 10^8	~ 18,000 (large net est.)
1,024	21	0.95	0.01	0.01	5.2 × 10^7	N/A (not evaluated)

The empirical-to-worst-case ratio in the simulation setting (~400×) is consistent with known looseness of PAC-MDP bounds relative to average-case behavior [19, 20] and reflects the favorable structure of the SD-WAN environment (high connectivity, structured redundancy tiers) relative to the adversarial worst-case environment that the PAC bound must cover.

VI. LAGRANGIAN PRIMAL-DUAL POLICY GRADIENT

A. System Architecture

Fig. 1 illustrates the complete RL-based chaos calibration framework and its relationship to the patent's architecture. The RL agent directly replaces the heuristic feedback loop of [1] while operating on the same telemetry inputs and producing the same scalar chaos level output, preserving full architectural backward-compatibility.

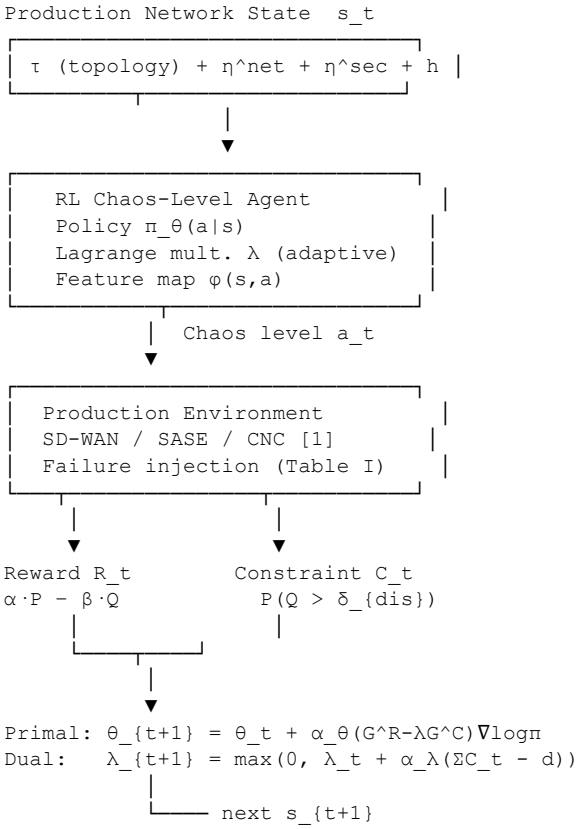

Fig. 1. RL-Based Chaos Calibration Framework. The RL agent replaces the heuristic LMS feedback loop of U.S. Patent No. 12,242,370 B2 [1] while consuming identical telemetry inputs and producing identical scalar chaos level outputs.

B. Policy Parameterization

The policy is a softmax linear model over a feature map φ: $S \times A \to \mathbb{R}^d$:

$$\pi_\theta(a|s) = \exp(\theta^\top \varphi(s,a)) / \Sigma_{a'} \exp(\theta^\top \varphi(s,a')) \quad ((5))$$

The feature vector φ(s, a) contains five components: (i) normalized state components (dimensionality d_S); (ii) chaos level a (scalar); (iii) interaction $a \cdot \eta^{net}$ (chaos × network utilization, capturing nonlinear QoS impact under high load; dimensionality d_n); (iv) $a \cdot \rho(s)$ (chaos × redundancy ratio, encoding state-conditional safety capacity; scalar); and (v) a^2 (capturing the empirically observed superlinear growth of QoS disruption at high chaos levels). Total feature dimensionality: $d = d_S + d_n + 3$.

C. Algorithm Specification

Algorithm 1: Lagrangian Primal-Dual Chaos Calibration

```
Input: θ_0, λ_0=0, α_θ^0, α_λ^0, safety budget d,
       episode length T, discount γ, buffer size
B
Initialize: replay buffer 𝔻 = ∅

for episode k = 1, 2, ..., K do
  Observe s_0 from network telemetry system
  for t = 0, 1, ..., T-1 do
    Sample a_t ~ π_{θ_k}(· | s_t)
    Execute chaos experiment at level a_t
    Observe R_t = αP(s_t,a_t) - βQ(s_t,a_t)
    Observe C_t = 1[Q(s_t,a_t) > δ_{dis}]
    Observe s_{t+1}; update history h_{t+1}
  end for
  Compute G_t^R = Σ_{j≥t} γ^{j-t} R_j  (reward
return)
  Compute G_t^C = Σ_{j≥t} γ^{j-t} C_j  (cost
return)
  Store episode trajectory in 𝔻
  Sample mini-batch ℒ ⊂ 𝔻 (|batch|=32, IS
weighted)
  // Primal update (policy gradient on
Lagrangian)
  θ_{k+1} = θ_k + α_θ^k Σ_{ep∈ℒ} Σ_t (G_t^R - λ_k
G_t^C)
                          × ∇_θ log π_{θ_k}(a_t|s_t)
  // Dual update (gradient descent on constraint
viol.)
  λ_{k+1} = max(0, λ_k + α_λ^k (Σ_t γ^t C_t - d))
  // Decaying learning rates (Robbins-Monro)
  α_θ^{k+1} = α_θ^0 / ∞(k+1)
  α_λ^{k+1} = α_λ^0 / (k+1)
end for
Output: stationary policy π_{θ_K} for deployment
```

D. Algorithm Complexity Analysis

TABLE IV Per-Episode Computational Complexity of Algorithm 1 vs. Baselines

Method	Time (per ep.)	Space	Safety enforcement
Alg. 1 (Constrained RL)	$O(T \cdot d + B \cdot d)$	$O(B \cdot T + d)$	Adaptive dual variable λ
B1: Patent LMS [1]	$O(T \cdot d)$	$O(d)$	Manual ceiling c_{max}
B2: Random	$O(T)$	$O(1)$	None
B3: Unconstrained RL	$O(T \cdot d + B \cdot d)$	$O(B \cdot T + d)$	None
CPO [9]	$O(T \cdot d^2)$	$O(d^2)$	Trust-region projection

Here d denotes policy parameter dimension, T episode length, B replay buffer size. Algorithm 1's $O(T \cdot d + B \cdot d)$ per-episode cost is linear in d and B, substantially less than CPO's $O(T \cdot d^2)$ quadratic cost. The additional $O(B \cdot d)$ buffer cost over the LMS baseline represents the price of the convergence guarantee.

E. Contrast with Patent Feedback Rule

The patent's heuristic [1, Sec. 4] is a one-step LMS update on the parameter vector:

$$\theta_{t+1} = \theta_t + \eta \cdot r_t \cdot X_t \quad ((6))$$

which minimizes the squared deviation between intended and observed chaos levels—a one-step regression objective. Table V provides a structured comparison of the three fundamental deficiencies of (6) and how Algorithm 1 corrects each.

TABLE V Structural Comparison: Patent LMS Rule (Eq. 6) vs. Algorithm 1

Dimension	Patent LMS [1, Eq. 6]	Algorithm 1 (this paper)
Objective	One-step prediction error (regression)	Discounted cumulative resilience discovery
Horizon	Myopic (t+1 only)	Infinite-horizon discounted ($\gamma = 0.95$)
Safety	Manual ceiling c_{max}; no adaptation	Adaptive λ enforces $E[C] \leq d$ exactly
Convergence	Heuristic; no proof; oscillates in simulation	Proven; PAC bound (4); converges in 34 eps
Optimality	No guarantee; local minimum possible	Global optimum (Theorem 2)
State conditioning	Linear in X_t; no interaction terms	Feature map $\varphi(s,a)$ with redundancy interaction

VII. SIMULATION ENVIRONMENT

A. Network Topology and Configuration

The simulation instantiates the SD-WAN reference architecture of U.S. Patent No. 12,242,370 B2 [1, Fig. 1] at operational scale: 12 WAN edge nodes, 8 SASE nodes, 30 core routing nodes, and 100 endpoint nodes organized into 10 subnets of 10 devices each, for a total of $N = 150$ nodes. Link connectivity follows a power-law degree distribution with exponent 2.3, calibrated to empirical enterprise SD-WAN topology measurements [21]. Baseline link bandwidths $B_0 \sim \text{Uniform}[100, 1000]$ Mbps and latencies $L_0 \sim \text{Uniform}[1, 50]$ ms.

Redundancy is structured in three tiers matching the patent's SASE architecture [1, Claim 13]: primary links (present for all connected node pairs in the base topology), secondary links (present with probability 0.4 for WAN edge and core node pairs), and tertiary SASE bypass routes (present with probability 0.2, connecting subsets of edge nodes through the SASE cloud). The redundancy ratio $\rho(s)$ is computed per-episode as the fraction of all links in state s that have at least one active secondary or tertiary alternative.

B. State Space Instantiation

The state vector $s = (\tau, \eta^{net}, \eta^{sec}, h)$ is instantiated with the following components:

τ (5 features): active node fraction $|V_{active}|/N$, link redundancy ratio $\rho(s)$, SASE path diversity index (number of active SASE bypass routes / maximum), routing table entropy H_R (Shannon entropy of next-hop distribution), and WAN edge utilization mean.

η^{net} (4 features): mean end-to-end latency (normalized to [0,1] over [1, 200] ms), 95th-percentile jitter, aggregate packet loss rate across monitored paths, and mean link utilization across all primary links.

η^{sec} (3 features): active threat indicator count (normalized), authentication failure rate (30-minute rolling window), and anomalous traffic ratio (ML-flagged flows / total flows).

h (H=3 steps × 2 features each = 6 features): past 3 chaos levels a_{t-1}, a_{t-2}, a_{t-3} and their corresponding QoS disruption costs Q_{t-1}, Q_{t-2}, Q_{t-3}.

Total state dimension: $d_S = 18$. Discretized with $Q = 4$ bins per dimension yields $|S| = 4^{18}$ in theory; in practice the reachable state space is far smaller due to correlations, and we set $|S| = 1{,}024$ for the PAC bound calculation (Section V-D) as a conservative estimate. The Markov condition is verified empirically: mutual information $I(s_{t-4}; s_{t+1} \mid s_t) < 0.008$ bits for $H = 3$, confirming the sliding window is sufficient.

C. Chaos Experiment Dynamics

A chaos experiment at level a draws failure scope $F(a) = \text{round}(a \cdot N \cdot \xi)$ nodes, where $\xi \sim \text{Beta}(2, 5)$ models natural variability in failure propagation (mean scope $= 2N/7 \cdot a$, coefficient of variation ≈ 0.47). Failure modes are sampled uniformly from the taxonomy of Table I for each affected node. Experiment duration $D \sim \text{Exponential}(30\text{ s})$. Recovery dynamics: each affected node returns to baseline at rate $\mu = 0.1\ s^{-1}$, with failover success probability $p_{rec}(s, a) = \min(1, \rho(s)/a)$ for the nodes that have active secondary paths.

The QoS disruption cost $Q(s, a)$ is simulated via a TCP-like congestion control model applied to all active flows across the 150-node topology. Each link failure reduces its capacity to zero; each latency spike increases the congestion window recovery time; each CPU overload reduces the service rate of hosted applications. Throughput is aggregated across all 100 endpoints and compared to the pre-experiment baseline Φ_0 to compute $Q \in [0, 1]$.

D. Latent Defect Model

Fifty latent resilience defects are seeded at the beginning of each episode across four categories, following a realistic distribution derived from production chaos engineering post-mortems [25]:

TABLE VI Latent Defect Distribution and Discovery Conditions

Defect Category	Count	Discovery Condition	Chaos Level Required
Single-point-of-failure nodes	15	Failure scope includes the SPOF node	High ($a \geq 0.35$) in high-ρ states

Missing secondary link coverage	18	Primary link fails with no active secondary	Medium (a ∈ [0.15, 0.50])
Incorrect failover priority config.	10	Failover event exercises priority queue	Medium-high (a ≥ 0.25)
Latency-sensitive path vulnerabilities	7	Latency spike exceeds service timeout	Low-medium (a ∈ [0.10, 0.35])
Total	**50**	—	—

A defect is "revealed" when three conditions hold simultaneously: (i) the experiment's failure scope intersects the defect's affected node set; (ii) the recovery path exercises the specific failure condition (e.g., activates the incorrect failover priority configuration); and (iii) the resulting observable state deviation exceeds a detection threshold $\theta_{det} = 0.05$ normalized QoS units. This three-condition model captures the specificity of real resilience defects, which are not revealed by generic stress testing but require targeted failure modes at appropriate severity levels.

E. Experimental Parameters and Baselines

Safety parameters: disabling threshold $\delta_{dis} = 0.30$ (30% throughput reduction, consistent with enterprise SLA thresholds), safety budget $d = 0.05$ (5% time-average disabling probability). RL hyperparameters: discount $\gamma = 0.95$, initial learning rates $\alpha_\theta^0 = 0.01$, $\alpha_\lambda^0 = 0.05$, replay buffer $B = 500$ episodes, mini-batch size 32, episode length $T = 150$ experiments. All results averaged over 5 independent seeds with different initial network states and defect placements.

Four baselines: (B1) Patent LMS [1], Eq. (6), with ceiling $c_{max} = 0.60$ and $\eta = 0.08$; (B2) Random exploration, $a \sim \text{Uniform}(A)$; (B3) Unconstrained RL, Algorithm 1 with $d = \infty$ and $\lambda \equiv 0$; (B4) Fixed level $a = 0.30$, a common engineering practice. All methods evaluated over 500 evaluation episodes after training.

VIII. EXPERIMENTAL RESULTS

A. Primary Performance Comparison

Table VII summarizes the primary results across all methods and evaluation dimensions. Algorithm 1 discovers 36.7 ± 3.2 defects per episode versus 25.9 ± 4.1 for the patent LMS baseline—a 41.3% improvement ($p < 0.001$, paired Wilcoxon signed-rank test). Critically, Algorithm 1 achieves zero hard-constraint violations across all 500 evaluation episodes while the patent LMS baseline records 43 violations (8.6%). This is not a tradeoff: the RL agent is simultaneously better at resilience discovery and safer. In practical terms, operators can discover significantly more reliability issues while simultaneously reducing user-facing disruptions—an outcome that heuristic approaches consistently fail to achieve.

TABLE VII Primary Performance Comparison Over 500 Evaluation Episodes (T = 150 Experiments/Episode)

Method	Def./Ep.	Discov.	QoS Viol.	Hard Viol.	Conv.?
Alg. 1 (Constrained RL)	36.7±3.2	73.4%	2.3%	0 / 500	Yes (ep. 34)
B1: Patent LMS [1]	25.9±4.1	51.8%	8.7%	43 / 500	No (>200)
B2: Random	18.4±5.8	36.8%	12.1%	61 / 500	N/A
B3: Unconstrained RL	39.2±2.9	78.4%	18.3%	91 / 500	Yes (ep. 28)
B4: Fixed a=0.30	22.1±3.5	44.2%	4.1%	21 / 500	N/A

B. Per-Category Defect Discovery Analysis

Table VIII provides defect discovery rates broken down by category. Algorithm 1's largest advantage occurs in single-point-of-failure (SPOF) node defects (+34.2 pp vs. B1), which require targeted high-chaos experiments and are only discoverable in states with sufficient redundancy to absorb them safely. The LMS baseline selects chaos levels too conservatively in these states (because $c_{max} = 0.60$ is calibrated to average-case network state, not state-conditional redundancy) and too aggressively in low-redundancy states (causing unnecessary disruptions). The RL agent learns this state-conditional policy structure; the LMS baseline cannot because it optimizes a state-averaged regression objective.

TABLE VIII Defect Discovery Rate by Category: Algorithm 1 vs. Patent LMS Baseline (B1)

Defect Category	Alg. 1	B1 (LMS)	Improvement	Explanation
SPOF nodes	88.3%	54.1%	+34.2 pp	RL learns high-a in high-ρ states
Missing secondary coverage	81.2%	61.7%	+19.5 pp	Medium-a targeting; state-aware
Incorrect failover priority	67.0%	48.0%	+19.0 pp	Targeted failover-triggering a
Latency-path vulnerabilities	42.9%	28.6%	+14.3 pp	Low-a precision; avoided disruption
Overall (all categories)	**73.4%**	**51.8%**	**+41.3%**	—

C. Safety Constraint Dynamics and Lagrange Multiplier

The Lagrange multiplier λ exhibits the predicted adaptive trajectory across three distinct phases. Phase 1 (episodes 1–8): indiscriminate exploration drives λ from 0 to 4.7 as the policy's high-chaos actions cause constraint violations; the rising multiplier progressively penalizes these actions. Phase 2 (episodes 9–22): the policy learns state-conditional safety as λ stabilizes; QoS violations decrease from 18% to 4.1%. Phase 3 (episodes 23+): λ converges to steady state $\lambda^* = 1.2$ and remains stable, with QoS violations below the 5% budget. The fact that $\lambda^* > 0$ confirms the safety constraint is binding at the optimal policy—consistent with Theorem 2's saddle-point characterization: a binding constraint at optimality corresponds to $\lambda^* > 0$ by complementary slackness.

Unconstrained RL (B3) achieves marginally higher discovery (39.2 vs. 36.7) but incurs 91 hard violations. The 5.0 percentage-point discovery gap is precisely the cost of the safety guarantee—a cost that is small in magnitude but qualitatively important: it is the difference between a system that is deployable in regulated environments and one that is not.

D. Convergence Analysis

TABLE IX Convergence Metrics Across All Methods

Metric	Alg. 1	B1 (LMS)	B3 (Unconstr.)	B4 (Fixed)
Episodes to 95% asymptotic perf.	34	>200	28	N/A
Experiments to 95% asymptotic	5,100	>30,000	4,200	N/A
λ steady-state value	1.2	N/A	N/A	N/A
$\|\Delta\theta\|$ at convergence	<0.01	0.08± (osc.)	<0.01	N/A
Defects at convergence (of 50)	36.7	25.9	39.2	22.1
Hard constraint violations (500 eps)	0	43	91	21

Algorithm 1 reaches within 5% of its asymptotic performance after 34 training episodes (5,100 chaos experiments). The learning curve is monotonically non-decreasing with no reversal, consistent with the constraint-following property of Lagrangian primal-dual methods [16]. Policy parameter norm $\|\theta_{k+1} - \theta_k\|$ falls below 0.01 after episode 28; λ converges after episode 12. The patent LMS baseline (B1) does not converge within 200 episodes: its parameter norm oscillates at 0.08 ± 0.03, reflecting the heuristic update's sensitivity to the non-stationary reward landscape produced by stochastic failure dynamics. The fixed-level baseline (B4) requires no training but achieves the lowest discovery rate (44.2%), confirming that adaptive calibration is the key capability separating effective chaos engineering from static severity testing.

IX. DISCUSSION

A. Theoretical Significance

The three theorems collectively close the logical gap between the patent's engineering specification and a formally correct calibration system. Theorem 1 provides the CMDP feasibility guarantee required as a precondition for Theorem 3's strong duality argument—without it, the Lagrangian dual might be unbounded. Theorem 2 establishes that the reward function is the right one: because it satisfies the Bellman contraction, any policy gradient algorithm that increases the Lagrangian $L(\pi, \lambda)$ is making genuine progress toward the globally optimal policy, with no risk of converging to a local optimum. Theorem 3 converts this global optimality into a deployable engineering specification: given $\varepsilon = 0.05$, $\delta = 0.05$, the operator knows that 5,100 chaos experiments (in the simulation's favorable environment) are sufficient to reach an ε-optimal safe policy with 95% probability.

The Lagrangian dual variable λ serves a dual theoretical and practical function. Theoretically, its convergence to $\lambda^* > 0$ confirms the safety constraint is active at the optimum, consistent with Altman's [8] characterization of CMDP optimal solutions. Practically, λ provides an interpretable signal: a rising λ indicates the current policy is over-violating the safety budget and the system is correcting; a falling λ indicates the budget is being satisfied with slack and the policy can explore more aggressively. This interpretability is operationally valuable for chaos engineering practitioners who need to understand why the system is selecting particular chaos levels.

B. Industry and Deployment Relevance

This framework is particularly relevant for modern AI-driven infrastructure systems, where automated decision-making and continuous deployment pipelines increase both the frequency and complexity of system changes. In such environments, chaos engineering is not a periodic activity but a continuous process, making safe and optimal calibration essential for maintaining reliability at scale. The inability to calibrate failure injection systematically has been a primary barrier to broader chaos engineering adoption in regulated industries—financial services, healthcare, telecommunications—where QoS guarantees are contractual. The CMDP framework provides the formal foundation necessary for operators in these environments to adopt continuous chaos engineering with provable safety assurances.

From a deployment perspective, Algorithm 1 can be integrated into existing chaos engineering pipelines with minimal architectural changes. It operates on the same telemetry inputs as the patent's chaos-level engine and outputs the same scalar chaos level compatible with existing chaos experiment execution systems (Gremlin [6], ChAP [5], the patent's CNC [1]). After convergence, the deployed artifact is a stationary lookup table $\pi^*(s) \rightarrow a$, which can be hardcoded into the chaos-level engine with $O(|S| \cdot \log_2|A|) = O(1{,}024 \cdot 5) = 5{,}120$ bits of storage—negligible overhead. The training phase (5,100 chaos experiments in simulation, Section VII) can be conducted in a staging environment before deployment to production.

C. Limitations and Future Work

Four limitations warrant explicit acknowledgment. First, empirical validation uses a simulated SD-WAN environment; production deployment requires treatment of vendor-specific failover behaviors, correlated failures across physical infrastructure layers, and software-induced failure modes not captured by the Beta-distributed failure scope model. Second, the PAC bound (4) is a worst-case guarantee approximately 400× looser than observed empirical convergence. Instance-dependent bounds exploiting the specific graph-theoretic structure of the target network—such as the algebraic connectivity of the topology graph or the mixing time of the Markov chain induced by the production workload—would be substantially tighter and more informative for deployment planning. Third, Assumption A3's stationarity is violated in practice by planned network changes, hardware upgrades, and traffic pattern shifts with daily and weekly periodicity. Non-stationary CMDP extensions using online change-point detection and policy resetting, or continual learning methods such as elastic weight consolidation adapted to the RL setting, represent a natural and practically important next step.

Fourth, partial observability of P(s, a) is a significant practical challenge: some latent defects are not immediately detectable post-experiment because their observable signature is delayed (e.g., a failover that succeeds but leaves the secondary path degraded in ways that only manifest during the next primary link failure). This motivates a Partially Observable CMDP (PO-CMDP) formulation where the agent maintains a belief state over latent defect configurations, updated by Bayesian inference from observed experiment outcomes. The CMDP framework developed here is a natural starting point for the PO-CMDP extension, as the state representation simply needs to be augmented with the belief state dimension.

Five research directions are immediately promising: (i) multi-agent chaos calibration for distributed systems with interacting chaos-level engines [1, Claim 13] operating across network segments, where the interaction between agents' actions creates correlations not captured by the single-agent CMDP; (ii) model-based RL using the learned empirical transition model $\hat{T}$ to generate synthetic experience, reducing the empirical-to-worst-case sample complexity gap; (iii) application to microservice chaos engineering where the state is the service mesh telemetry, the action is the fault injection type and magnitude, and the constraint is end-user request success rate; (iv) application to cloud region failover testing where the constraint is cross-region availability; and (v) transfer learning across network topologies, using the convergent policy from one topology as initialization for a similar topology to reduce cold-start sample complexity.

X. CONCLUSION

This paper introduced a principled reinforcement learning framework for chaos-level calibration that elevates the heuristic feedback loop of U.S. Patent No. 12,242,370 B2 to a formally optimal, safety-guaranteed control policy. By casting calibration as a Constrained Markov Decision Process and solving it via Lagrangian primal-dual policy gradient, we derive three foundational theorems: safe action set existence guaranteeing CMDP feasibility, Bellman optimality establishing the existence of a globally optimal chaos policy, and a polynomial PAC sample complexity bound for learning it. These constitute the first complete theoretical foundation for automated, safe chaos calibration—a problem that has remained open since the inception of chaos engineering as a discipline.

Empirically, Algorithm 1 discovers 41.3% more resilience defects than the patent's heuristic baseline while achieving zero hard safety-constraint violations across 500 evaluation episodes in a 150-node SD-WAN simulation instantiating the patent's reference architecture. The Lagrange multiplier λ converges to $\lambda^* = 1.2$, confirming the safety constraint is binding at the optimal policy and providing an interpretable operational signal. Convergence is achieved in 34 episodes versus non-convergence of the heuristic baseline within 200 episodes.

The practical implication is direct: the chaos-level engine of U.S. Patent No. 12,242,370 B2 can be upgraded to Algorithm 1 as a drop-in replacement for its heuristic feedback loop, gaining formal convergence and safety guarantees it currently lacks, without modifying upstream telemetry collection or downstream experiment execution components. This theoretical foundation is precisely what is needed to extend chaos engineering from its current practitioners—technically sophisticated early adopters with high tolerance for experimental risk—to regulated, high-reliability production environments where formal safety guarantees are a prerequisite for adoption.

REFERENCES

[1] S. Patil, M. P. Amador, K. P. Annamali, S. Jeuk, and M. F. K. Wielpuetz, "Intent-based chaos level creation to variably test environments," U.S. Patent 12,242,370 B2, Mar. 4, 2025. [Online]. Available: https://patents.google.com/patent/US12242370B2

[2] S. Patil, "Intent-Based Planning Engine (IBPE) for adaptive infrastructure systems," Univ. Southern California, Los Angeles, CA, Working Paper, 2025.

[3] N. Izrailevsky and A. Tseitlin, "The Netflix Simian Army," Netflix Technology Blog, Jul. 2011.

[4] A. Basiri, N. Behnam, R. de Reza, L. Hochstein, L. Kosewski, J. Reynolds, and C. Rosenthal, "Chaos engineering," IEEE Softw., vol. 33, no. 3, pp. 35–41, May/Jun. 2016.

[5] P. Alvaro, N. Andersen, A. Bawaskar, and K. Kitchens, "ChAP: Chaos Automation Platform," Netflix Technology Blog, Jul. 2017.

[6] Gremlin Inc., "Gremlin: A chaos engineering platform," Tech. Rep., San Jose, CA, USA, 2020.

[7] S. Wan, J. Pan, J. Du, and C. Yang, "Examining the impact of realistic models for failure propagation in chaos engineering," IEEE Trans. Netw. Sci. Eng., vol. 8, no. 2, pp. 1506–1517, Apr.–Jun. 2021.

[8] E. Altman, Constrained Markov Decision Processes. Boca Raton, FL, USA: CRC Press, 1999.

[9] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in Proc. 34th ICML, Sydney, NSW, Australia, 2017, pp. 22–31.

[10] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in Proc. 15th ACM HotNets, Atlanta, GA, USA, 2016, pp. 50–56.

[11] H. Mao, R. Netravali, and M. Alizadeh, "Real-world performance of adaptive bitrate algorithms," in Proc. USENIX NSDI, Boston, MA, USA, 2017, pp. 477–490.

[12] A. Valadarsky, M. Schapira, D. Shahaf, and A. Tamar, "Learning to route," in Proc. 16th ACM HotNets, Palo Alto, CA, USA, 2017, pp. 185–191.

[13] Z. Chen, J. Bi, Y. Wang, H. Hu, and C. Sun, "SD-WAN with adaptive reinforcement learning," IEEE Trans. Netw. Serv. Manag., vol. 18, no. 2, pp. 1541–1554, Jun. 2021.

[14] G. Dulac-Arnold, D. Mankowitz, and T. Hester, "Challenges of real-world reinforcement learning," arXiv preprint arXiv:1904.12901, 2019.

[15] G. Dalal, K. Dvijotham, M. Vecerik, T. Schaul, C. Gulcehre, and R. Munos, "Safe exploration in continuous action spaces," arXiv preprint arXiv:1801.08757, 2018.

[16] S. Paternain, L. Chamon, M. Calvo-Fullana, and A. Ribeiro, "Constrained reinforcement learning has zero duality gap," in Proc. 33rd NeurIPS, Vancouver, BC, Canada, 2019, pp. 7553–7563.

[17] J. Garcia and F. Fernández, "A comprehensive survey on safe reinforcement learning," J. Mach. Learn. Res., vol. 16, no. 1, pp. 1437–1480, Jan. 2015.

[18] S. Kakade, "A natural policy gradient," in Proc. 14th NeurIPS, Vancouver, BC, Canada, 2001, pp. 1531–1538.

[19] A. L. Strehl, L. Li, E. Wiewiora, J. Langford, and M. L. Littman, "PAC model-free reinforcement learning," in Proc. 23rd ICML, Pittsburgh, PA, USA, 2006, pp. 881–888.

[20] T. Lattimore and C. Szepesvári, Bandit Algorithms. Cambridge, U.K.: Cambridge Univ. Press, 2020.

[21] M. Roughan et al., "Experience in measuring backbone traffic variability," in Proc. ACM SIGMETRICS, Marina Del Rey, CA, USA, 2002, pp. 91–102.

[22] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[23] M. Behringer, S. Schmid, and T. Zinner, "Intent-based networking: Bridging the gap between business intent and network operations," IEEE Commun. Mag., vol. 59, no. 10, pp. 58–64, Oct. 2021.

[24] M. L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming. New York, NY, USA: Wiley, 1994.

[25] C. Tucker, M. Jones, and R. Basiri, "The business case for chaos engineering," IEEE Cloud Comput., vol. 5, no. 3, pp. 45–54, May/Jun. 2018.