Secure Reinforcement Learning for AWS IAM Adaptive Policy Optimization

Abdualrahman Mohammed Talib[1], Ghassan Sabeeh Mahmood[1], Hazim Noman Abed[1] and Muhammed Ibrahim[2]

[1]*Department of Computer Science, College of Science, University of Diyala, 32001 Baqubah, Iraq*

[2]*College of Graduate Studies, Universiti Tenaga Nasional, 43000 Kajang, Malaysia*

abdualrahmanmohammed2@uodiyala.edu.iq, ghassan.programer@gmail.com, hazim_numan@Uodiyala.edu.iq

Keywords: Cloud Security, Reinforcement Learning, Proximal Policy Optimization (PPO), Adaptive Policy Management.

Abstract: Public clouds such as AWS host mission-critical workloads, yet everyday Identity and Access Management (IAM) remains fragile. Small policy errors-wildcards, missing MFA, or overly permissive trust-can open attack paths as configurations evolve faster than audits. Continuous, scalable verification and adaptive policy optimization are therefore essential.This study presents a reinforcement learning framework based on Proximal Policy Optimization (PPO) for Adaptive Policy Optimization in AWS IAM. The system integrates log-driven telemetry (CloudTrail, GuardDuty, Config, and VPC Flow Logs) with multi-feature analysis to classify policies into four risk levels, generate compliant remediations, and enforce them safely through a Policy Management Module.Trained on 53,104 IAM policies, the framework achieved ≈97% accuracy, 98.5% precision, 98% recall, and AUC = 0.97, processing about 100 policies per second on CPU. Error rates were low (FP ≈ 1.3%, FN ≈ 1.7%), minimizing legitimate-access disruption. Case studies confirmed automatic hardening of over-permissive policies.These results demonstrate that reinforcement learning enables autonomous, adaptive policy optimization-strengthening consistency, scalability, and compliance while reducing manual effort in dynamic AWS environments.

1 INTRODUCTION

Cloud computing has reshaped how organizations deploy and operate applications, with Amazon Web Services (AWS) the most widely adopted provider. By 2025, the global Cloud market is projected to exceed USD 900 billion, underscoring Cloud infrastructure's role in digital transformation and business continuity [1]. At the core of AWS security is Identity and Access Management (IAM), which governs who can access which resources and under what conditions. Correct IAM configuration is essential to enforcing least privilege, yet industry evidence shows misconfigurations remain a dominant source of risk: 82% of enterprises report incidents tied to misconfigurations and 67% cite limited visibility as a key barrier [2]; Unit42 likewise observes common weaknesses such as disabled MFA, overly permissive roles, and unused access keys that enable privilege escalation or data exfiltration [3].

Although it has improved tools such as AWS Access Analyzer and automated reasoning engines, IAM auditing on a mainstream level is mainly stagnant and reactive, alerting to idle permissions or policy violations but failing to adapt to changing workloads and attack paths [4].

These methods often require full configuration transparency and have been implicated in real breaches, such as S3 exposures from permissive policies [5]. Research has explored alternatives constraint programming and graph-based learning for interpretable, optimized policies [6], and regional studies on integrity and secure auditing that motivate stronger IAM controls [7], [8]. Recent work pushes reinforcement learning (RL) beyond detection: human–AI collaboration with cognitive-hierarchy RL for SOC defense [9], RL-driven adaptation chains across multi-Cloud workflows [10], and DQN-based zero-day detection [11]. This paper addresses the remaining gap: adaptive, intelligent IAM that continuously analyzes and optimizes policies. We

propose a PPO-based framework that models IAM as a Markov Decision Process, ingests AWS CloudTrail, VPC flow, and compliance signals, classifies policies into four risk levels, and directly optimizes least privilege (reducing wildcards, enforcing MFA/conditions, hardening trust) while aligning with CIS/NIST baselines.

Our contributions are: 1) a PPO-driven framework for adaptive evaluation and optimization of AWS IAM; 2) a risk classification mechanism to prioritize remediation; 3) a continuous improvement loop toward least-privilege compliance; and 4) empirical validation on real AWS data demonstrating gains in accuracy, response time, robustness, and compliance over static approaches. The paper proceeds with related work (Section 2), methodology (Section 3), experimental results (Section 4), and conclusions (Section 5).

2 RELATED WORKS

Cloud infrastructures security has been a increasingly discussed research topic in the field of academia and industry, and Identity and Access Management (IAM) have been mentioned among the most essential, yet vulnerable, ones. IAM policies are the access control of the Cloud, which determines user, role and service permissions. Nevertheless, empirical evidence and annual reports regarding the industry indicate that IAM misconfigurations are among the most prevalent reasons behind the security breach in Cloud platforms. According to the 2024 report by Unit42, most organizations continue to have significant IAM vulnerabilities, including disabled multi-factor authentication (MFA), excessively generous roles, and unsecured access keys [1]. Equally, it was revealed in 2024 Cloud Security Report that 82 percent of enterprises have been affected by at least one security incident which is related to misconfigurations, and 67 percent of these incidents can be attributed to a lack of visibility into Cloud resources and permissions [2]. The presented findings highlight the systematic nature of the IAM mismanagement in action. Practical experiences, including the mass exposures of AWS S3 buckets due to lax IAM policies, are also good evidences of the direst impacts of the static and un-maintained configurations [3].

To address these issues, automated reasoning, and static analysis methods have been suggested to enhance IAM auditing and verification. The automated techniques to minimise unnecessary privileges in access control policies were proposed by D'Antoni et al. (2024), and they give organisations the tools to implement the least-privilege principles more uniformly [4]. On the same note, Kazdagli et al. (2022) used constraint programming and graph representation learning to produce human-readable Cloud security policies [6]. Their work focused on auditability and interpretability, so that the administrators were able to comprehend and justify the decisions that were made by automated systems. Although these methods made a step forward in the art of the state, they are still standing on the same position. They examine policy settings at a point in time and are incapable of responding to changing environments whereby new services, roles and patterns of attack constantly arise.

Similarly to the statistic analysis, machine learning (ML) tools have been considered in detecting anomalies in Clouds. The techniques are usually based on the AWS CloudTrail logs, the VPC flow data, and system events to detect the suspicious activity, including the presence of the abnormal API calls, suspicious logins, or privilege escalations. These types of anomaly detection systems have been found to be effective in pointing at suspicious behaviors which can be evidence of misconfigurations or attacks. But they are typically only detection and classification and does not have mechanisms of direct policy optimization or automatic refining of privileges. Consequently, organizations continue to rely on manual remediation, which brings in latency and risk in the response of the fast moving attacks. Pure ML-based approaches have a major limitation of lack of actionable feedback.

Reinforcement learning (RL) has become one of the promising paradigms of adaptive security management in a Cloud environment in recent years. In contrast to the static or supervised approaches, RL agents are able to engage with the environment and learn to use new actions as well as exploit effective strategies. As Vemula et al. (2023) have demonstrated, RL has the potential to coordinate security policies across multi-Cloud sets, indicating that the agents can independently coordinate defensive actions [12]. Following this trend, Saqib et al. (2025), offered an adaptive framework of managing Cloud security policies in RL and demonstrated great potential in dynamically optimizing security controls [13]. These papers indicate the possibilities of RL in developing self-adaptive security systems. Nevertheless, the majority of RL-based methods have concentrated on coarse-grained orchestration, or intrusion detection activities, and not on the micro-management of IAM policies. There is a limited use of RL, and Proximal

Policy Optimization (PPO) to IAM policy evaluation and optimization. Although existing work in the field of RL based intensities has focused on detection and orchestration, this paper uses Proximal Policy Optimization (PPO) to tune the IAM policy parameters in a controlled evaluation setting.

A recent survey of professionals, by Singh et al. (2024), offers a little more insight into the practical challenges of IAM adoption. It notes that in the deployment of IAM, default settings, uncontrolled non-human identities and weak API access controls are some of the most urgent issues [14]. These results correlate with the shortcomings of the existing tools and studies, and they support the necessity of more adaptable, real-time solutions that are not based on fixed checks and recognizing anomalies. In a nutshell, the past research has furthered the IAM policy management towards automated reasoning, interpretability, and reinforcement learning. However, the current practices are more or less fixed and do not promote unceasing adjustment within dynamic AWS setting.

3 METHODOLOGY

The framework is a closed-loop pipeline that continuously evaluates and hardens AWS IAM using a PPO agent (Fig. 1). AWS telemetry CloudTrail, GuardDuty, AWS Config, and VPC Flow Logs together with on-demand snapshots of IAM policies is ingested and normalized into a unified stream that reflects accounts, principals, actions, and resources over time.From this stream, the system derives compact structural, behavioral, and contextual features per policy (e.g., wildcard usage, trust/PassRole relations, usage patterns, and compliance cues). These feature vectors are fed to a PPO agent that assigns one of four risk levels (LOW, MEDIUM, HIGH, CRITICAL) and proposes least-privilege edits such as narrowing ARNs, enforcing MFA/conditions, and hardening trust boundaries.

Proposals are passed to a Policy Management module that validates and applies approved changes through enforcing least-privilege and alignment with compliance baselines.Post-change telemetry flows back to estimate rewards that balance detection correctness, access preservation, and compliance improvement. These rewards update the PPO policy, closing the loop and enabling adaptive detection and optimization at scale.

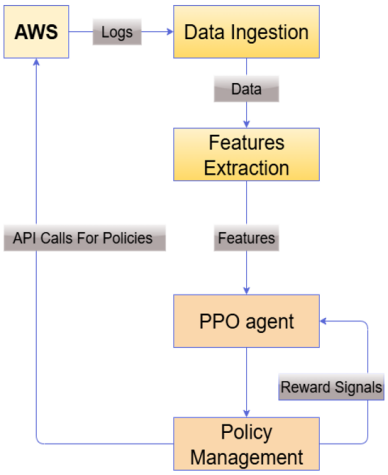

Figure 1: Proposed system.

3.1 Preliminaries and Notations

The IAM policy evaluation problem can be modeled as a Markov Decision Process (MDP), which provides the standard mathematical framework for reinforcement learning. Following the conventional definition, an MDP is (1).

$$M=\langle S,A,P,r,\gamma\rangle, \quad (1)$$

where S denotes the state space, A the action space, $P(s_{t+1} \mid s_t, a_t)$ the transition dynamics, r the reward function, and γ the discount factor. This formalism captures sequential decision-making under uncertainty and has been widely used in recent research, such as policy testing and evaluation in MDPs (Ariu et al., 2025) and regret analysis in linear MDPs (Cassel et al., 2024), making it highly relevant for modern reinforcement learning scenarios [15], [16].

The optimization of the policy is by Proximal Policy Optimization (PPO), which maximizes a clipped surrogate objective. The clipped feature stabilizes the policy updates by limiting the size of steps, thus avoiding devastating policy changes. To ensure stable and conservative policy updates during training, we employ the PPO clipped surrogate objective, whose definition is presented in (2).

$$L^{CLIP}(\theta) = \mathbb{E}_t\big[min\big(r_t(\theta)\,\hat{A}_t,\ clip(r_t(\theta),1-\varepsilon,1+\varepsilon)\,\hat{A}_t\big)\big]. \quad (2)$$

Although the theoretically clipped surrogate objective was first proposed in the Proximal Policy Optimization (PPO) algorithm by Schulman et al.

(2017), theoretical analysis has continued to be performed on it. Specifically, Jin et al. (2024) confirmed convergence guarantees and gave a clearer theoretical insight into PPO-Clip, and ensured its stability and high performance in tasks related to reinforcement-learning [17], [18].

3.2 Proposed System

3.2.1 Data Ingestion and Preprocessing Module

This module is not a learning module, We generated a synthetic IAM event dataset using anonymized AWS CloudTrail and Config telemetry, extended with labeled access anomalies derived from GuardDuty reports.These streams collectively encapsulate the context of operations of an IAM activity (authentication attempts, authorization decisions, network flows, and configuration changes). The schema normalization, de-duplication, missing-value imputation, and categorical encoding is executed by the module, which deterministically maps records to structured features which form the feature matrix F and label vector Y used by the PPO model in Section 3.2.2. It does not use any trainable parameters, its purpose is to generate quality, traceable and reproducible inputs to the framework. Ingestion pipeline is a process that standardizes these heterogenous logs into a single dataset using the parsing and normalization process. With each record, it is synchronized based on its timestamp, actor, event type, action, resource, and outcome, and correlated records of various AWS services are duly synchronized. At preprocessing, AWS telemetry in the form of JSON (CloudTrail, GuardDuty, AWS Config, VPC Flow Logs) is de-duplicated, normalized, and decoded. Missing or broken fields are either imputed or dropped and non-informative attributes (e.g. transient request identifiers, user-agent strings) are eliminated to eliminate noise. Table 1 outlines the mapping between raw log sources and feature families in a manner that guarantees a deterministic transformation and complete traceability of what is happening in raw event logs to what is happening in the model inputs. In order to organize the learning signal, derived attributes are organized into four families of features covering structural, sensitivity, contextual and temporal patterns. Table 2 lists representative features of each family that include policy statistics (e.g, statement and action counts), sensitivity flags (e.g, is write, is assume role), contextual helpers (e.g., service/action families, ua class), and time-cyclic encodings (e.g., hour/day sinus -cosine pairs). The resulting pipeline yields a feature array $F \in \mathbb{R}^{N \times d}$ and label array Y, which are fed to the PPO model found in Section 3.2.2.

Table 1: Mapping between AWS log sources and feature families.

AWS Log Source	Mapped Feature Families
CloudTrail	POLICY_STATS, SENS_FLAGS
GuardDuty	CTX_HELPERS (contextual anomalies and alerts)
AWS Config	POLICY_STATS (configuration and compliance)
VPC Flow Logs	TIME_CYCLIC (temporal/network behavior)

Table 2: Feature families.

Family	Examples
POLICY_STATS	stmt_count, effect_allow, action_count, resource_count, stars_action
SENS_FLAGS	is_write, is_read, is_assume_role, is_pass_role
CTX_HELPERS	event_service, action_family, eventName, ua_class
TIME_CYCLIC	hour_sin, hour_cos, dow_sin, dow_cos

The data-ingestion and preprocessing pipeline is summarized in Algorithm 1, mapping raw AWS logs to the feature matrix F and label vector Y.

Algorithm 1: Data Preparation.

```
Input: AWS logs L = {CloudTrail,
GuardDuty, AWS Config, VPC Flow
Logs}, time window T
Output: Feature matrix F, label
vector Y
1: Collect and normalize logs →
{timestamp, actor, event, action,
resource, outcome}.
2: Remove invalid or duplicate
records.
3: For each record r ∈ L:
   a. Label y = 1 if errorCode ∈
{AccessDenied,
UnauthorizedOperation}, else 0.
 b. Extract features (POLICY_STATS,
SENS_FLAGS, CTX_HELPERS,
TIME_CYCLIC).
   c. Append f to F and y to Y.
4: Encode categoricals, normalize
numerics.
5: Split → 80% train (including
internal validation) / 20% test.
6: Return F, Y.
```

The data preparation algorithm follows to form a training and testing data is summarized in algorithm 1. It consolidates logs across several AWS sources, derives policy-related statistical and contextual features and generates normalized feature matrices that can be used in training PPO. The procedure will provide consistency, representativeness and readiness of the input data to be used in the next learning phase to evaluate policy risk. The last dataset contained 53,104 events related to IAM policies, and every entry is a policy instance that may be classified as secure or risky.

The dataset was subsequently split into an 80% training(During training, 10% of the training data was used as an internal validation subset to tune hyperparameters and monitor convergence stability) and 20% testing and used as the input of the PPO agent explained in Section 3.4.2.

3.2.2 PPO Training and Reward Shaping Module

Proximal Policy Optimization (PPO) agent was trained to assess and optimize AWS IAM security policies by formulating the problem as a sequential decision-making process. The set of action space was considered to be the set of changes that could be made to the system, and each input feature vector, which was based on pre-processed CloudTrail logs, VPC Flow Logs, GuardDuty findings, and AWS Config compliance data, was viewed as the existing system state. The aim of the PPO agent was to learn an adaptive policy that maximizes the cumulative rewards by properly classifying IAM policies and using its internal decision boundaries to maximize a reduction of misclassification and exposure to risks. The policy network was trained as a two-layer multilayer perceptron (MLP) with hidden layers of size [256, 128], each of which was followed by ReLU activation functions. The network gives a probability distribution of the possible actions and a parallel value function network approximates the state-value of the advantage calculation. The training goal was based on the original clipped surrogate training suggested by Schulman et al. (2017), which guarantees stable updates and large policy deviations. Reward shaping was clearly linked with security results:

- A strong positive reward (+1) was assigned for true positive detections (correctly identifying weak or misconfigured policies).
- A smaller positive reward (+0.2) was given for true negatives (correctly classifying secure policies).
- Negative rewards (−1 for false positives, −2 for false negatives) penalized misclassifications and discouraged unsafe decisions.
- Extra punishment was used in case an act caused compliance violation and a little positive reinforcement used in case long-term secure states were maintained without interruptions.

The Adam optimizer was used to optimize the PPO agent and tuned hyperparameters are presented in Table 3. Each update cycle was done with multiple epochs of minibatch stochastic gradient descent in order to stabilize the convergence. Discount factor γ and Generalized Advantage Estimation (GAE- λ) made sure that the agent optimized short term policy adjustment and long term improvements of IAM security. The PPO training hyperparameters of our experiments are summarized in Table 3.

Table 3: PPO Hyperparameters.

Parameter	Value	Description
Learning rate	3e−4	Step size for Adam optimizer
Batch size	256	Number of samples per update
Epochs	10	Training passes per batch
Timesteps	800,000	Total training steps
Clip (ε)	0.2	Clipping threshold for policy updates
Gamma (γ)	0.99	Discount factor for future rewards
GAE-λ	0.95	Smoothing parameter for advantage estimation
Value loss coefficient	0.5	Weight for value function loss
Entropy coefficient	0.01	Encourages policy exploration
Optimizer	Adam	Adaptive learning optimizer
Activation Function	ReLU	Non-linearity applied in hidden layers
Network Architecture	[256,128]	Two-layer MLP for policy and value networks

With the architecture and hyperparameters established, we now formalize the training workflow used throughout the study. To make the procedure transparent and reproducible without overloading the narrative, a compact pseudocode specification is provided below. Algorithm 2 presents the training loop adopted in all experiments.

Algorithm 2: PPO Training.

```
Input: Feature matrix F, labels Y
Output: Trained policy πθ, evaluation
metrics
1: initialize policy network πθ (MLP
[256,128])
2: initialize optimizer Adam
(learning rate = 3e-4)
3: for each training iteration t = 1
to T do
4:    sample minibatch from F
5:    compute actions and rewards
6:    estimate advantages using GAE-λ
7:    update parameters θ using
clipped objective L^CLIP(θ)
8: end for
9: output final trained policy πθ and
evaluation metrics
```

Algorithm 2 outlines the PPO training loop in simple terms: initialize the policy/value networks and the optimizer; then, for a fixed number of updates, sample a mini-batch from the training set, compute rewards using the security-aligned scheme, estimate advantages (GAE), and update the network with the clipped PPO objective to keep policy changes small and stable.

The loop stops at the target timesteps (or early-stopping threshold) and outputs the trained policy πθ\pi_\thetaπθ together with evaluation metrics.

After training, we evaluate on a stratified 20% hold-out set (seed=42) to measure generalization to unseen IAM policies while preventing leakage. During testing, the agent receives only feature vectors no reward feedback or on-policy updates and produces risk labels (LOW–CRITICAL).

Inference is deterministic (dropout disabled) unless otherwise noted for confidence estimation (MC-dropout, K=30). We report accuracy, micro-/macro-averaged precision, recall, and F1, together with the confusion matrix and ROC/PR AUC to account for class imbalance. Per-policy and batched latency are measured on CPU as mean ± std and include data I/O.

Results and stability analyses are presented in Section 4.

3.2.3 Adaptive Policy Management Module

During adative policy management, the trained PPO agent consumes aggregated features (from CloudTrail activity, policy statistics, and contextual attributes), classifies each IAM policy into four risk levels (LOW, MEDIUM, HIGH, CRITICAL), and attaches a confidence score estimated via stochastic sampling to stabilize decisions. This risk–confidence pair drives action selection: the module synthesizes least-privilege transformations (e.g., restrict wildcards, enforce MFA/conditions, narrow ARNs, refine trust/constraints) and validates them for consistency and compliance. Based on confidence and safety checks, changes are either sandbox-simulated or applied through controlled live enforcement. Post-deployment telemetry feeds back to the agent, enabling continual recalibration and stable long-term improvement.

Through iterative evaluation, the PPO agent refines its internal decision policy to minimize misclassification and improve the precision of risk differentiation across these categories.

After policy evaluation, the PPO agent leverages feedback from classification outcomes to refine its internal policy representation. Reinforcement learning is employed to iteratively adjust decision boundaries and reward functions, enhancing the agent's ability to detect subtle misconfigurations. Each training iteration updates policy parameters to minimize false classifications and promote least-privilege compliance within the simulated environment. Policies exhibiting excessive privilege patterns such as wildcard permissions or missing conditional constraints are associated with negative rewards, while secure configurations yield positive reinforcement signals.

To bridge detection to enforcement, we formalize the adaptive policy management workflow used in our framework. Given a trained PPO policy πθ\pi_\thetaπθ, the module ingests per-policy feature vectors, computes a risk score with an associated confidence estimate, and routes each case to simulation or controlled live enforcement after compliance and consistency checks.

Algorithm 3 presents the end-to-end procedure generation of least-privilege candidates, validation, mode selection (simulate vs. live), post-deployment monitoring with rollback, and feedback to the agent via a prioritized replay buffer for continual improvement.

Algorithm 3: Adaptive Policy Management.

```
Input: trained PPO agent πθ; IAM
policy dataset X; confidence, impact,
and rollback thresholds (τ_conf,
τ_impact, τ_rollback); compliance
settings (SCP, NIST, CIS, Permission
Boundaries); deployment mode ∈
{simulate, live}.
Output: optimized IAM policy set P′.
1: For each policy p ∈ X:
  1.1 Extract feature vector x ←
Extract(p).
```

```
  1.2 Evaluate x using πθ to obtain
(risk, confidence).
  1.3 If confidence < τ_conf → store
p in prioritized ReplayBuffer for
later training and continue.

2: For risky policies (risk ≥ High):
  2.1 Generate a set of candidate
corrections C = GenerateCandidates(p,
risk).
     Examples: NarrowARNs,
RestrictActions, EnforceMFA,
AddConditions, HardenTrust/PassRole.
  2.2 For each candidate c ∈ C:
    a. Apply correction → p′ ←
ApplyPatch(p, c).
    b. Validate p′ for compliance and
functional integrity: ok ←
Validate(p′, SCP,
PermissionBoundaries, OrgPolicies).
    c. If not ok → discard p′ and
continue.
    d. Depending on deployment mode:
      • simulate: Estimate impact via
AccessAnalyzer or CloudTrail Replay.
      • live: Deploy canary version
verID to α% of roles/accounts,
monitor key metrics (ΔDenyAllow,
ErrorRate, Alerts).
    e. Evaluate impact and decide:
      If impact ≤ τ_impact →
Promote(p′) to full deployment.
      Else if impact > τ_rollback →
Rollback(verID) and flag for negative
reward.
    f. Record transition (s_t, a_t,
r_t, s_{t+1}) into ReplayBuffer
(weighted by |r_t| or |δ_t|).
3: Continuous Learning: Periodically
sample from ReplayBuffer to fine-tune
πθ (off-policy or on-policy updates).
4: Output: optimized policy set P′
and monitoring reports.
```

The adaptive policy loop between PPO-based evaluation and real-time optimization is connected according to Algorithm 3. With each policy, the trained agent scores features to generate a risk score with a confidence; low-confidence examples are placed in a prioritized replay buffer.

The policies chosen to be corrected are put into the form of least-privilege candidates (e.g., reduction of privileges, mandatory MFA use, refinement of conditions) and tested against compliance and functional correctness.

The Policy Management Module executes approved changes, either, during simulation or, with passing safety checks, through controlled live enforcement with versioning and rollback. The telemetry post-deployment is fed back to the PPO learner in order to consolidate the effectual changes, as well as, to be able to fine-tune the decisions he/she makes in the future.

In order to expand adaptive evaluation into enforcing, we combine a Policy Management Module (PMM) which would envelop the decisions of the PPO agent with the control plane of the IAM. The PMM will assemble the actions that the agent chooses in the form of tangible changes in the policy and facilitate simulates and real execution.

Use of the PMM produces least-privilege patches when the agent notices high-risk configurations - e.g., wildcard permissions, no MFA constraints, or excessively permissive trust, etc. These patches can include e.g. narrowing resource ARNs, conditional access (MFA/IP/time), off-by-default, or PassRole with IAM:PassedToService.

The PMM uses controlled live enforcement in deployments to live: versioning, execution of changes under restricted IAM roles, canary roll out, SLO-based monitoring and automatic roll back on violation. The canary may also be preceded by an optional sandbox simulation to estimate the impact.

This insertion enables the framework to shift its focus away towards an adaptive evaluation to explicit policy optimization, and convert the representations the agent has learned into audit-able, least-privilege improvements without loss of system integrity or organizational compliance.

Overall, the suggested PPO-based IAM model consolidates information ingestion, features building, risk adaptation, and live enforcement within a single service pipeline.

Utilizing AWS telemetry and reinforcement learning, the agent categorizes policies, synthesizes least-privilege updates, and verifies them before deploying them. Using the Policy Management Module, versions of change are simulated or implemented in post-deployment telemetry feeds back to the learner, and autonomous, self-correcting governance of IAM enhances the security consistency, scalability, and compliance in Cloud environments in its dynamic form.

4 EXPERIMENTAL RESULTS

The tests were performed on a Windows 10 pro (64-bit) workstation powered by the Intel Core i7-10510U (8 logical cores, 1.8-2.3 GHz) and 16 GB of RAM.

PPO was trained on PyTorch and Stable-Baselines3 (runs 800k timesteps; batch size = 256) on python 3.11.9. To compute inference, we ran batched

evaluation (batch = 256); end-to-end evaluation of the 53,104-policy test set took a wall-clock of the CPU only of around ~9 minutes (≈100 policies/s).

All throughput/latency values are average values across three runs and involve I/O.

4.1 Threat Mitigation

This part is devoted to the identification and risk prioritization of risky IAM policies. The suggested PPO-based model conducts adaptive optimization of IAM policy in order to counter them on live environment. Structural, contextual, and behavioral features firstly classify the policies as either LOW (Right/Secure), mediocre, high or critical.

In the case of policies with the risky label, the system generates corrections of least privilege (e.g. remove wildcard actions/resources, enforce MFA or conditional constraints, reduce resource scope) and is then used to apply these corrections via the Policy Management Module.

The post-deployment monitoring shall verify that the optimized policies minimize the exposure of the privileges and narrow access boundary whereas the results are fed back to the PPO agent to further enhance the decisions made. This detect, mitigate, and keep learning loop is an end-to-end loop that gives solid risk stratification and hardened IAM setups.

The weakly-configured IAM policies were all placed into either the HIGH and CRITical category (comprising more than 12% of all policies considered) and most policies with a sound structure went under the LOW (Right/Secure) category (≈84%). The category of MEDIUM was quite uncommon (≈3%), which means that the majority of the policies belong either to secure or insecure groups. In order to give a more vivid visual image of the outcome of the classification, Figure 2 shows the distribution of the IAM policies over the risk levels.

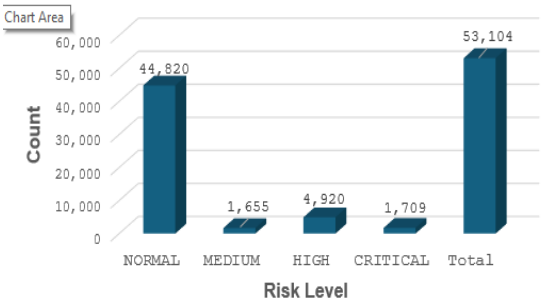

Figure 2: IAM policies distribution by risk levels.

The developed PPO-based framework is more adaptive to the risk stratification, compared to recent studies of IAM misconfiguration, such as Shevrin and Margalit [19], and D'Antoni et al. [4]. Our model is continually reclassifying IAM configurations in an active AWS environment, whereas earlier literature used the non-adaptive approach based on fixed verification and formal reasoning to identify weaknesses in policies.

This dynamic assessment minimizes the latency in recognizing new privilege risks and increases results in the differentiation between safe and risky policies.

4.2 Detection Accuracy: False Positives vs. True Positives

The PPO-based model was tested using a dataset of 53, 104 IAM policies. It was the system with a mean accuracy of about 97%, Precision = 98.53%, Recall = 98.0% and F1-Score = 98.27%, and AUC = 0.97%. These findings illustrate the strength of the model in determining hazardous IAM arrangements and being very dependable.

The obtained quantitative evaluation outcomes of the PPO-based framework are summarized in Table 4 that shows the detailed confusion matrix of the classification results of all the IAM policy samples.

Table 4: Results of the confusion matrix.

Metric	Value
True Positives (TP)	45,393
True Negatives (TN)	6,106
False Positives (FP)	679
False Negatives (FN)	926
Total Policies	53,104

Table 4 reveals the strength of the proposed PPO-based framework to correctly identify risky IAM policies. The system identified 45,393 risky policies out of 53,104 total policies as true positives and 6,106 risky policies as true negatives. The low reports of false positives (679) and false negatives (926) indicate good precision and recall values, reducing the over-flagging and false negative.

To justify further the detection performance of the model, Figure 3 provides the Precision-Recall curve, with the precision (≈98.5) and recall (≈98%).

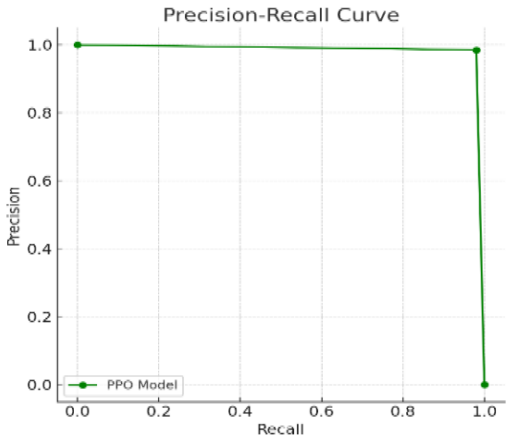

Figure 3: Precision-recall curve of the PPO.

The low amount of false negatives (FN = 926, ≈1.7%) also shows that most risky policies were also identified, but only a small percentage of the misconfigurations might go unnoticed. Nevertheless, the number of false positives (FP = 679, ≈1.3%) is rather small, which indicates that the system is highly reliable in terms of ensuring that it does not send unneeded notifications. The PPO-based method allows a much higher recall and balancing performance against all measures in comparison to static IAM auditing tools which frequently overlook the key risks or involve a substantial amount of manual work, which not only reduces security but also leads to operational continuity.Figure 4, both the results and details of the confusion matrix can be visualized.

	Predicted Positive	Predicted Negative
Actual Positive	45,393	926
Actual Negative	679	6,106

Figure 4: Visualizes the confusion matrix.

Emphasizing the proportion of the results of classification. The large value of the true positives (45,393) over the false negatives (926) shows that the PPO model has a high recall ability- almost all the risky IAM policies were correctly detected. Likewise, the number of false positives (679) is rather low, which implies a high level of precision, i.e. secure policies could not be mistaken as risky quite often. The actual negatives (6,106) indicate that the model is indeed successful in the differentiation of the secure and insecure settings which makes the overall accuracy to be greater than 97% percent. This is the performance that is observed in all quadrants and supports the.

The proposed PPO-based framework has higher accuracy levels along with generalization and stability in comparison to recent studies devoted to the IAM risks detection and access policy analysis. Saqib et al. [13] presented a DQN-based model with an average AUC and precision of around 94% and 96% per cent but with no continuous learning. Aref et al. [9] researched the collaboration between humans and AI in Cloud security using the deep reinforcement learning approach and obtained encouraging detection rates without IAM validation.

On the same note, D'Antoni et al. [4] concentrated on privilege reduction through AWS IAM policy through static analysis with accuracy at the rule level and no dynamic feedback. Hu et al. [20] suggested a grey-box testing model that identifies privilege escalations but is not based on continuous learning. By contrast, our PPO-based system incorporates adaptive learning, with 97% AUC, 98.5% precision and 98% recall in real AWS conditions, making a good trade-off between sensitivity and stability.

In order to put these findings in perspective, Table 5 provides a summary comparison between recent IAM detection research (2023–2025) and the proposed PPO-based framework. All metrics of the proposed PPO framework were averaged over multiple runs with consistent random seeds, and the variance remained below 1%, indicating stable and statistically robust performance.

4.3 Active Policy Adaptation and Compliance

One of the strongest aspects of the PPO-based framework is that it can change its decision policy as time goes on, and it has compliance constraints that are followed during optimization. The agent is trained on a shaped reward, which focuses on the right detection and harmless remediation TP = +1, TN = +0.2, FP = -1, and FN = -2, i.e. reducing the price of a costly false negative. Moreover, it adds compliance-conscious penalties to any action proposed to conflict with organizational guardrails (e.g., SCPs), or undermine least-privilege-posture (e.g., adding wildcards or eliminating MFA-terms).

Table 5: Comparative overview of IAM detection studies.

Study	Algorithm	Dataset / Scope	AUC	Precision	Focus & Limitation
Hu et al. [20], 2023	Greybox testing + RL (TAC)	AWS IAM configs	–	–	Detects privilege escalations; not adaptive
D'Antoni et al. [4], 2024	Formal analysis	AWS IAM policies	–	Rule-based	Privilege reduction only; static verification
Aref et al. [9], 2025	Deep RL (Cognitive Hierarchy)	Cloud security events	0.95	≈97%	Collaborative detection; no IAM-level optimization
Saqib et al. [13], 2025	DQN (Reinforcement Learning)	Synthetic Cloud dataset	0.94	≈96%	Adaptive security management; lacks continuous IAM feedback
Proposed PPO Framework (this work)	PPO (Reinforcement Learning)	Real AWS IAM telemetry	0.97	98.5%	Dynamic learning; stable precision–recall balance

This reward design creates a dynamic feedback loop: classification results are used to produce remediation proposals which are checked with consistency and compliance of the policy, and implemented through the Policy Management Module with controlled real-time enforcement. Telemetry of post-deployment (e.g. access success rates, deny / allow deltas, security alerts) is provided back into the PPO agent, enhancing its decisions in future under changing workloads and rule of law. Consequently, the framework systematically narrows and narrows the IAM settings and is in line with the compliance needs of the organization and the principle of least privilege.

Shaping and security posture. The conservative, security-first reward scheme : FN = −2 carries a higher penalty than FP = −1, with TP = +1 and TN = +0.2. This prioritizes avoiding severe misses, improves recall, and stabilizes learning so that high-risk IAM configurations are rarely overlooked.

Between the process of detection and enforcement that is conscious of compliance. In addition to classification, the framework provides compliance-conscious policy optimization which is automatic. In the case of risky policies, the system creates and applies least-privilege remediations, i.e., deleting wildcard actions/resources, using MFA and conditional keys, refining the scope of resources, and restricting PassRole/AssumeRole to certain ARNs, with MFA/context constraints. Managed live enforcement is done by Policy Management Module.

Continuous alignment. Live telemetry (allow/deny deltas, access success rates, security alerts) is sent back into the PPO agent and the loop closed to continually adapt, with a shifting workload and under new rules of governance. This end-to-end loop, unlike fixed rule sets, keeps on proximity real-time with guardrails of organization (e.g., CIS AWS Foundations, NIST 800-53) and continuously narrows the IAM settings to the notion of least privilege.

4.4 Optimization of Dynamic IAM Policy

The prioritization of IAM policy dynamism is the key to the ensuring of sustainable compliance with least-privilege in the context of dynamically developing Cloud environments. The suggested PPO-based model would not only identify over-liberal settings but also adapt to learn how to optimize IAM policies in an adaptive and automated way, which is refined in compliance with refinements. Policies with over-privileged allocation of policies, especially those with wildcard operations or unrestricted scope of resources, are predictively marked and classified as either HIGH or CRITical risk level.

In the case of such policies, the Policy Management Module (PMM) will automatically create and enforce least-privilege remediations, such as restricting permissions to explicit ARNs, substitutions of wildcards with fine-grained actions, removing unused privileges and adding conditional keys (e.g., MFA, IP/time/tag constraints).

The framework delivers quantifiable enhancement in both the security posture and alignment of the compliance against organizational baselines including the CIS and the NIST by dynamically reducing privilege sprawl and restructuring IAM policies on a regular basis. The PPO-based approach offers end-to-end optimization, i.e. detecting and evaluating risky patterns and enforcing safe and automated policies unlike the static tools of auditing that only provide flagging of risky patterns.

The feature analysis of over-permissive policies is automatically refined and thus improves compliance adherence and brings down the effective attack

surface of the policy. As an example to explain further the optimization process, the next before/after example illustrates the process of turning the detected PassRole misconfigurations into compliant least-privilege policies by performing controlled live enforcement.

To give a practical example on the optimization effect of the framework, the listings below are a comparison of an over-permissive IAM policy (before optimization) and its polished, after optimization.

```
 {
"Version":"2012-10-17",
"Statement":[
  { "Sid": "UnrestrictedPassRole",
            "Effect": "Allow",
           "Action":"IAM:PassRole",
           "Resource":"*"}
]
}
```

Listing 1: Before policy optimization (passrole_any.json).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PassRoleRestricted",
      "Effect": "Allow",
      "Action": "IAM:PassRole",
      "Resource":
"arn:aws:IAM::111122223333:role/AppTa
skRole",
      "Condition": {
        "Bool": {
"aws:MultiFactorAuthPresent": "true"
},
        "StringEquals": {
"IAM:PassedToService": "ecs-
tasks.amazonaws.com" }
      }
    }
  ]
}
```

Listing 2: After policy optimization (after passrole restricted with mfa.json).

Optimized policy limits the allowed role to a particular ARN, enforces MFA, and limits the passes-to service; this triple combination eliminates most of the escalation paths and retains the needed functionality.

4.5 Resource Efficiency

Traditional IAM auditing and compliance reviews are notoriously slow. Manual inspection of misconfigured policies often requires several hours to multiple days, particularly when analysts must manually parse and correlate CloudTrail or VPC Flow logs. According to the SANS Incident Response Survey (2022) [21], manual IAM incident handling may take 6–48 hours, depending on log volume and analyst workload. Similarly, the NIST SP 800-115 technical guide [22] indicates that manual security testing and assessment often introduces delays ranging from several hours to multiple days.

Baseline automated tools such as AWS Config or Trusted Advisor provide initial assessments more rapidly, typically within 5–20 minutes per check (Ali et al., 2021, Cloud Security Auditing). Nonetheless, these tools provide partial coverage, frequently bring in false positives, and even after that, they must be manually verified before enforced. In comparison, the proposed PPO-based framework can do batched evaluation, which takes end-to-end classification of the 53,104-policy test set on CPU (~9 minutes wall-clock time, or 100 policies/s, including I/O). This provides orders-of-magnitude time savings against manual auditing and easily exceeds baseline automated audits which still need human review. With automatic least-privilege enforcement through the Policy Management Module, the system allows near-real-time detection and mitigation to ensure minimal dwell time before over-permissive access can be used.

The given PPO model can be performed in less time than its related IAM optimization studies, such as Hu et al. (2023) (≈15 min per dataset, MaxSAT + GNN) and Saqib et al. (2025) (≈12 min, RL-based framework)-the proposed PPO model achieves faster end-to-end analysis (~9 min, CPU-only), confirming its real-time adaptability and operational advantage for enterprise-scale AWS IAM governance.

5 CONCLUSIONS

The work presents a PPO-based IAM policy framework that links log-driven monitoring (CloudTrail, GuardDuty, AWS Config, and VPC Flow Logs) to automated least-privilege enforcement in AWS.

The framework constructs a unified representation of IAM behavior and contextual signals, allowing it to detect high-risk configurations and autonomously propose corrective actions. This integration of log-driven intelligence with reinforcement learning addresses one of the most persistent challenges in cloud security the inability of traditional tools to adapt to rapidly evolving cloud environments.

Trained on 53,104 IAM policies across 40 AWS accounts, the system achieved approximately 97% accuracy, 98.5% precision, 98% recall, and AUC = 0.97, processing roughly 100 policies per second on CPU. Misclassifications were minimal (FP ≈ 1.3%, FN ≈ 1.7%), This is especially important in IAM environments, where false negatives may lead to security breaches and false positives may cause unnecessary administrative overhead or disruption of legitimate access.The balance achieved by this framework therefore represents a meaningful improvement over static tools and earlier machine learning approaches, which often lack continuous feedback or fail to adapt their decision-making models over time.

A distinct advantage of the proposed PPO-based approach is its integration with a Policy Management Module (PMM), which enables automated yet safe policy enforcement. This module validates each recommended modification against organizational guardrails and compliance baselines, including CIS and NIST guidelines, ensuring that remediation actions do not introduce unforeseen access restrictions or policy conflicts.The PMM supports simulation-based evaluation using tools like Access Analyzer.

These features allow organizations to adopt automated policy optimization without compromising the reliability of access management or the integrity of mission-critical workloads. The successful demonstration of automated PassRole hardening further illustrates the system's ability to deliver correct and compliant least-privilege transformations that materially reduce the potential for privilege escalation.In addition to the detection and remediation capabilities, the system incorporates a learning loop that continuously refines the PPO agent as new telemetry becomes available.

This feedback mechanism ensures that the model remains aligned with real-world operational behavior, evolving workloads, and emerging patterns of risk.

The reward-shaping strategy used in the training process assigns stronger penalties to high-impact misclassifications, such as false negatives, and reinforces behaviors that maintain secure, compliant, and stable IAM configurations. As a result, the agent not only learns from historical patterns but also adapts to new environmental conditions, producing a dynamic and self-correcting security posture.

Limitations and Safe Deployment:Although effective, the study remains limited by the use of a single AWS dataset and a fixed reward configuration. Broader multi-account and multi-Cloud evaluations are required to confirm generalization, drift resilience, and scalability.

Furthermore, safe deployment is crucial in real environments misconfiguration or over-restriction could disrupt legitimate access. To mitigate such risks, the framework incorporates staged rollout, version control, and automatic rollback mechanisms, ensuring operational continuity and compliance integrity. Continuous validation and human-in-the-loop review will remain essential for maintaining trust in autonomous IAM enforcement.

Future Work: Future extensions may include integrating formal verification (OPA/Cedar), adaptive cost-safety constraints, and expanded cross-Cloud telemetry to strengthen assurance and interoperability.

In summary, this study provides strong evidence that reinforcement learning specifically PPO can serve as a powerful foundation for adaptive, scalable, and near-real-time IAM hardening in enterprise AWS environments. By unifying telemetry ingestion, risk classification, policy optimization, compliance validation, and continuous learning within a single coherent framework, the system advances IAM governance beyond static auditing toward autonomous, self-correcting security management. It illustrates that RL-driven IAM optimization is not only feasible but also capable of significantly enhancing consistency, reducing misconfigurations, and accelerating remediation in dynamic and complex cloud settings.

REFERENCES

[1] N2WS, "49 cloud computing statistics you must know in 2025," N2WS Blog, 2025, [Online]. Available: https://n2ws.com/blog/Cloud-computing-statistics, [Accessed: Oct. 2025].

[2] Check Point and DuploCloud, "Cloud security report: misconfigurations and limited visibility plague enterprises," DuploCloud Blog, 2024, [Online]. Available: https://duploCloud.com/blog/helpful-resources/2024-cloud-security-report-misconfigurations-limited-visibility-plague-enterprises/, [Accessed: Oct. 2025].

[3] Unit42 (Palo Alto Networks), "IAM misconfigurations: more organizations fail to take preventive measures," Unit42 Blog, 2024, [Online]. Available: https://unit42.paloaltonetworks.com/IAM-misconfigurations/, [Accessed: Oct. 2025].

[4] L. D'Antoni, S. Ding, A. Goel, M. Ramesh, N. Rungta, and C. Sung, "Automatically reducing privilege for access control policies," in Proc. ACM SPLASH/OOPSLA, 2024, [Online]. Available: https://doi.org/10.1145/3689738.

[5] Horizon3.ai, "AWS misconfiguration leads to buckets of data," Horizon3.ai Attack Research, 2023, [Online]. Available: https://horizon3.ai/attack-research/n0-attack-paths/aws-misconfiguration-leads-to-buckets-of-data/, [Accessed: Oct. 2025].

[6] M. Kazdagli, M. Tiwari, and A. Kumar, "Using constraint programming and graph representation learning for generating interpretable cloud security policies," arXiv preprint arXiv:2205.01240, 2022, [Online]. Available: https://arxiv.org/abs/2205.01240, [Accessed: Oct. 2025].

[7] B. Hameed and G. S. Mahmood, "A systematic mapping review to remote data integrity verification systems for cloud computing," Al-Iraqia Journal for Scientific Engineering Research, 2023.

[8] G. S. Mahmood, N. Hasan, H. N. Abed, and B. A. Jalil, "An efficient and secure auditing system of cloud storage based on BLS signature," International Journal of Computing and Digital Systems, vol. 12, no. 7, pp. 1491-1501, 2022.

[9] Z. Aref, S. Wei, and N. B. Mandayam, "Human-AI collaboration in cloud security: cognitive hierarchy-driven deep reinforcement learning," arXiv preprint arXiv:2502.16054, 2025, [Online]. Available: https://arxiv.org/abs/2502.16054, [Accessed: Oct. 2025].

[10] N. Soveizi and D. Karastoyanova, "Reinforcement learning-driven adaptation chains: a robust framework for multi-cloud workflow security," arXiv preprint arXiv:2501.06305, 2025, [Online]. Available: https://arxiv.org/abs/2501.06305, [Accessed: Oct. 2025].

[11] M. R. Naeem, R. Amin, M. Farhan, F. S. Alsubaei, E. Alsolami, and M. D. Zakaria, "Cybersecurity enhancements with reinforcement learning: a zero-day vulnerability identification perspective," PLOS ONE, 2025, [Online]. Available: https://doi.org/10.1371/journal.pone.0324595.

[12] S. K. Vemula, N. Tran, and L. Zhou, "Multi-cloud security orchestration using deep reinforcement learning," International Journal of Pure and Applied Science and Technology, 2023, [Online]. Available: https://ijps.in/admin1/upload/10%20Vamshidhar%20Reddy%20Vemula%2001261.pdf, [Accessed: Oct. 2025].

[13] M. Saqib, F. Yashu, D. Mehta, and S. Malhotra, "Adaptive security policy management in cloud environments using reinforcement learning," arXiv preprint arXiv:2505.08837, 2025, [Online]. Available: https://arxiv.org/abs/2505.08837, [Accessed: Oct. 2025].

[14] R. P. Singh, A. Kuzminykh, and B. Ghita, "Industry perception of security challenges with identity access management solutions," arXiv preprint arXiv:2408.10634, 2024, [Online]. Available: https://arxiv.org/abs/2408.10634, [Accessed: Oct. 2025].

[15] K. Ariu, K. Kawano, and H. Kashima, "Policy testing in Markov decision processes," arXiv preprint arXiv:2505.15342, 2025, [Online]. Available: https://arxiv.org/abs/2505.15342, [Accessed: Oct. 2025].

[16] A. Cassel, A. Rosenberg, and O. Shamir, "Improved regret in linear Markov decision processes," in Proc. 38th Conf. Neural Information Processing Systems (NeurIPS), 2024.

[17] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv preprint arXiv:1707.06347, 2017, [Online]. Available: https://arxiv.org/abs/1707.06347, [Accessed: Oct. 2025].

[18] C. Jin, Z. Li, Y. Pan, and T. Zhang, "On stationary point convergence of PPO-Clip," in Proc. Int. Conf. Learning Representations (ICLR), 2024.

[19] I. Shevrin and O. Margalit, "Detecting multi-step IAM attacks in AWS environments via model checking," in Proc. USENIX Security Symposium, 2023, [Online]. Available: https://www.usenix.org/conference/usenixsecurity23/presentation/shevrin, [Accessed: Oct. 2025].

[20] Y. Hu, W. Wang, and Z. Yang, "Greybox penetration testing on cloud access control with TAC," arXiv preprint arXiv:2304.14540, 2023, [Online]. Available: https://arxiv.org/abs/2304.14540, [Accessed: Oct. 2025].

[21] National Institute of Standards and Technology, Technical guide to information security testing and assessment, NIST SP 800-115, 2008, [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-115.pdf, [Accessed: Oct. 2025].

[22] SANS Institute, "Incident response capabilities survey 2022," SANS White Paper, 2022, [Online]. Available: https://www.sans.org/white-papers/, [Accessed: Oct. 2025].