Proceedings of Machine Learning Research 299:1–24, 2025 Conference on Applied Machine Learning for Information Security

RoleSentry: A Multi-Stage Framework for Explainable Detection of AWS Role Chaining Attacks

Godwin Attigah GODWIN@GODWINATTIGAH.COM
Independent Research, NJ, USA

Austin Gansz AUSTIN.GANSZ@GMAIL.COM
Independent Research, NC, USA

Editor: Edward Raff and Ethan M. Rudd

Abstract

AWS AssumeRole enables essential automation but also provides attackers with a primary mechanism for privilege escalation and lateral movement. Security teams face two critical challenges: identifying suspicious individual `AssumeRole` events within overwhelming legitimate activity and detecting when sequences of seemingly benign events form malicious role-chaining attacks. RoleSentry addresses both challenges through a novel three-stage framework combining behavioral filtering, ensemble anomaly detection, and graph neural networks.

The system first applies a lightweight behavioral trait filter that automatically suppresses routine service-to-service automation, resulting in a 12.3% volume reduction with zero conflicting classifications. Filtered events undergo parallel analysis: an ensemble model scores individual events using contextual features, while a graph neural network analyzes temporal graphs to detect multi-step chaining patterns. The system provides SHAP-based explanations for analyst interpretation.

We evaluated RoleSentry on a large corpus of `AssumeRole` events collected from over 30 days. Compared to Amazon GuardDuty, RoleSentry reduced false positive alerts by 98% and identified 25 sophisticated attacks that the commercial service missed entirely, including complex role-chaining scenarios. The results demonstrate that combining behavioral filtering with graph-based analysis provides operationally viable detection of both isolated and chained `AssumeRole` abuse.

Keywords: Machine Learning, Anomaly Detection, Cloud Security, AWS

1. Introduction

In 2024, Invictus, a security consulting company, investigating an incident at a customer site uncovered a sophisticated attack pattern that highlights a critical vulnerability in cloud security monitoring (Invictus Security, 2024). The researchers named the incident 'DangerDev' after discovering that attackers had gained initial access to the customer's AWS Infrastructure through compromised administrator credentials. The attacker then established persistent backdoor access by creating IAM roles with carefully crafted trust policies. The trust policies allowed assumption from external AWS accounts controlled by the attackers. Attackers repeatedly used the AssumeRole API from their rogue accounts to re-enter the victim environment, demonstrating how role assumptions serve as a primary mechanism for maintaining covert access and performing lateral movement (Invictus Security, 2024; Amazon Web Services, 2024d; Unit 42, Palo Alto Networks, 2024).

As AWS maintains the largest market share in the global cloud computing industry (Dinsdale and Staff, 2025), the AWS Identity and Access Management (IAM) service is paramount for a vast number of organizations worldwide, as it fundamentally defines the security landscape of any AWS implementation. **AWS IAM AssumeRole** stands as a foundational part of AWS IAM, as it enables the delegation of granular permissions across services, accounts, and users, fostering automation and secure privilege separation when a principal calls `AssumeRole`(Amazon Web Services, 2024e). While indispensable for AWS operations, `AssumeRole` has become a primary attack vector for adversaries who seek to escalate privileges, conduct lateral movement, and establish persistent access. Attackers consistently leverage role assumptions to achieve critical objectives following initial compromise. Attacks against IAM abuse the `AssumeRole` mechanism as a primary method for escalating privileges in AWS environments (MITRE Corporation, 2024c,b; Amazon Web Services, 2025c). After assuming a role, both legitimate and malicious actors consequently use the **AWS Security Token Service (STS)** to retrieve temporary security credentials that grant access with permissions defined by the assumed role. Numerous high-profile incidents demonstrate the severity of such abuse, including the Capital One breach, where attackers exploited SSRF vulnerabilities to facilitate massive data exfiltration, and the SCARLETEEL campaign, which involved extensive lateral movement through containerized applications (Sources and Reports, 2019; Team, 2023).

Attackers often employ role chaining when seeking alternative critical paths within the environment (Elastic, 2024; Netskope Research Team, 2021; Shevrin and Margalit, 2023). They iteratively assume multiple roles to piece together the permissions required for their objectives, progressively expanding access and deepening their foothold within AWS environments (Levine and Sonntag, 2021; CrowdStrike, 2024b). Organizations often exacerbate the problem when legitimate operational requirements for cross-account access and service integration create overly permissive trust relationships. Attackers exploit both necessary trust configurations and outright misconfigurations, including unrestricted cross-account role assumptions and flaws in core AWS services (Team, 2022; Labs, 2021; Security, 2020; Labs, 2023; Splunk Threat Research Team, 2021; Orca Security Research Team, 2023; Sysdig Research Team, 2023). The continuous, high volume of legitimate operational activity, such as tens of millions of daily AssumeRole events from AWS services and CI/CD pipelines (Amazon Web Services, 2024b; Datadog, 2024; Sysdig, 2022), creates a severe signal-versus-noise problem for Enterprises, particularly their Security Operations Centers (SOCs) (Tariq et al., 2024; Experts, 2024). Within this overwhelming volume of activity, security teams *face two critical challenges*:

1. Assessing whether an individual AssumeRole event deviates meaningfully from expected behavior.

2. Detecting when a sequence of AssumeRole events, each benign in isolation, collectively forms a malicious chain.

Existing security products and services, including Amazon GuardDuty, struggle to address both challenges effectively (Skyhawk Security, 2024; CloudOptimo, 2025b; Virtu, 2023; Amazon Web Services, 2024c). These systems generate overwhelming volumes of false positives from benign operational changes and provide opaque explanations that fail to

clarify why events were flagged as suspicious, leading to analyst fatigue (Tariq et al., 2024; AWS Community, 2024). Furthermore, prior research utilizing model-checking techniques for detecting multi-step IAM attacks primarily focuses on static policy verification and pre-deployment analysis (Shevrin and Margalit, 2023; Prizmant et al., 2022), identifying potential vulnerabilities in configurations rather than detecting actual, real-time exploitation of AssumeRole through analysis of operational event data. These static analysis tools, focused on policy and configuration, are unable to detect the complex, evolving attack sequences that manifest as behavioral patterns in streaming event logs from live AWS environments.

Given the challenges above, a shift towards a more intelligent, context-aware, and transparent approach is necessary to identify and flag multi-step attack patterns explicitly. RoleSentry directly addresses the significant challenges and offers a robust solution, presenting a novel framework with the following key contributions.

RoleSentry makes the following key contributions to the field of cloud security:

- Designed a behavioral trait filter that identifies and suppresses benign automation across AssumeRole activity.
- Developed an ensemble classifier that scores individual AssumeRole events using contextual and identity-based features.
- Constructed a graph representation of suspicious AssumeRole events and applied a graph neural network to detect multi-step role chaining behavior.
- Provided SHAP-based local explanations and graph visualizations to support analyst interpretation of alerts.
- Evaluated RoleSentry on 500 million AssumeRole events and compared its performance to Amazon GuardDuty in both isolated and chained attack detection.

The subsequent section provides essential background on AWS IAM and CloudTrail. The foundation proves crucial for understanding the data and problem space RoleSentry addresses.

2. Background: IAM and CloudTrail

The section provides essential context for understanding the detection of suspicious AssumeRole activity. It delves into AWS identity management fundamentals (Spacelift, 2024; Amazon Web Services, 2025b), the mechanics of the AssumeRole workflow (Amazon Web Services, 2024e), and how attackers exploit these systems through role chaining (Elastic, 2024; Netskope Research Team, 2021; Shevrin and Margalit, 2023; Orca Security, 2024; SentinelOne, 2024).

2.1. Fundamental IAM Concepts

AWS Identity and Access Management (IAM) operates on a default-deny model where policies explicitly grant access(Amazon Web Services, 2025b). IAM's core components create an access control system that, when misconfigured, enables powerful attack vectors (Spacelift, 2024). These components include:

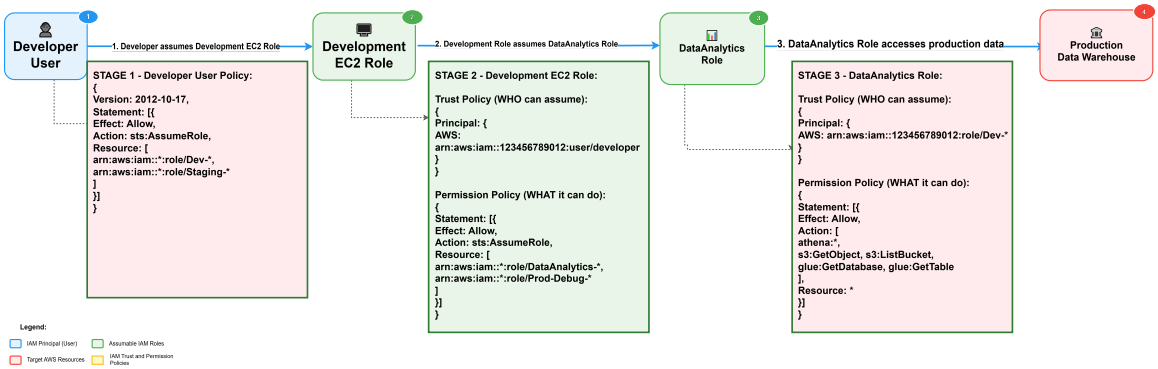

Figure 1: Multi-hop AssumeRole attack pattern showing how individually constrained IAM policies create unintended privilege escalation paths. Developer credentials (Stage 1) chain through Development EC2 Role (Stage 2) to access production data (Stage 3).

Table 1: Summary of the evaluation dataset characteristics.

Metric	Value
Time Period	30 Days
Total AssumeRole Events	540,290,797
Distinct AWS Accounts	1,894,077
Distinct Assumed Roles (ARNs)	13,146,381
Distinct Calling Principals	1,894,077
Distinct Source IP Addresses	164,917

- **Principals:** Entities make requests to AWS services. Principals include IAM Users, IAM Roles, and AWS Services (e.g., S3 for Datastore and SageMaker for ML/AI Workflows). In Figure 1, the '(Developer User)' and the '(Development EC2 Role)' both act as principals.
- **Policies:** JSON documents define permissions, including trust policies that control role assumption. For example, the trust policy for the '(Development EC2 Role)' in Stage 2 of Figure 1 specifies which principals can assume it.
- **Security Token Service (STS):** This service manages AssumeRole operations and issues temporary credentials.

Attackers frequently exploit overly permissive policies, such as those granting permission to assume any role in the AWS account or organization (Team, 2022; Labs, 2021; Security, 2020; Labs, 2023; Splunk Threat Research Team, 2021; Orca Security Research Team, 2023; Sysdig Research Team, 2023). While role chaining itself is a legitimate operational capability, these types of policies enable attackers with initial credentials to chain through multiple roles, escalating privileges and accessing resources they should not access (Elastic, 2024; Netskope Research Team, 2021; Shevrin and Margalit, 2023; Levine and Sonntag, 2021; CrowdStrike, 2024b).

2.2. The AssumeRole Workflow and Attack Patterns

The `AssumeRole` operation enables temporary privilege delegation through the following workflow (Amazon Web Services, 2024e), as conceptually illustrated in Figure 1:

1. **A principal calls the AssumeRole API with its current credentials.** This action initiates the process, akin to the '(Developer User)' initiating AssumeRole in Stage 1 of Figure 1.
2. **STS validates the request against the target role's trust policy.** This validation ensures that the calling principal has permission to assume the specified role, a crucial security check before granting temporary credentials.
3. **STS returns temporary credentials (access key, secret key, session token).** These credentials grant the principal the permissions of the assumed role for a specified duration.
4. **The principal uses these temporary credentials for subsequent API calls.** This enables the principal to perform actions allowed by the assumed role, as seen with the '(Development EC2 Role)' performing actions in Stage 2 and the '(DataAnalytics Role)' accessing data in Stage 3 of Figure 1.

AWS CloudTrail records every AssumeRole call as an individual atomic event, creating the audit trail RoleSentry analyzes (CloudOptimo, 2025a; Amazon Web Services, 2025a; Sysdig, 2022; CrowdStrike, 2024a; Edge Delta, 2025). Table 2 shows the key fields for security analysis that RoleSentry leverages.

Table 2: Critical `AssumeRole` CloudTrail fields for security analysis.

Field	Security Significance
`userIdentity.type`	The field indicates if the caller uses temporary credentials (`AssumedRole`), a key for detecting chaining.
`sourceIPAddress`	Geographic anomalies often reveal compromised credentials.
`requestParameters.roleArn`	The value identifies the target role in the chain, revealing the attack path.
`eventTime`	Temporal patterns distinguish automation from attacks.

2.3. Role Chaining Attack Mechanics

Role chaining occurs when attackers iteratively assume multiple roles to achieve their objectives (Elastic, 2024; Netskope Research Team, 2021; Shevrin and Margalit, 2023; Orca Security, 2024; SentinelOne, 2024). While role chaining is a legitimate mechanism for privilege delegation in AWS, attackers can abuse it to escalate privileges and move laterally within an environment (Levine and Sonntag, 2021; CrowdStrike, 2024b). Figure 1 illustrates a common multi-stage role chaining process when exploited by an attacker. The attack progression typically follows these steps, as depicted in the diagram:

1. **Developer Assumes Development EC2 Role (Stage 1 in Figure 1):** A developer, acting as an IAM Principal, first assumes a `Development EC2 Role`. This initial assumption provides an entry point into the AWS environment, often through vectors discussed in our threat model (e.g., compromised credentials or insider threat). The IAM policy for the '(Developer User)' in Figure 1 grants permission for this initial assumption.

2. **Development Role Assumes DataAnalytics Role (Stage 2 in Figure 1):** The `Development EC2 Role` then assumes a `DataAnalytics Role`. This action represents a lateral movement, where the attacker leverages the permissions of the first assumed role to gain access to a second, often more privileged, role. The '(Development EC2 Role)' in Figure 1 shows a trust policy allowing this assumption.

3. **Data Analytics Role Accesses Production Data (Stage 3 in Figure 1):** Finally, the `DataAnalytics Role` accesses production data stored in a Production Data Warehouse. This step demonstrates the ultimate objective of the role chain: accessing sensitive resources. The '(DataAnalytics Role)' in Figure 1 has a permission policy allowing access to production data.

The sequence highlights a significant detection challenge: each individual step may appear legitimate in isolation, but the sequence of assumptions reveals malicious intent when observed in context (Shevrin and Margalit, 2023; Orca Security, 2024).

Threat Model. RoleSentry's threat model addresses individual suspicious `AssumeRole` events and focuses on key scenarios where attackers leverage `AssumeRole` operations, including specific multi-step role chaining patterns:

1. **Credential Theft:** Stolen keys from developer endpoints enable systematic role enumeration (MITRE Corporation, 2024a).

2. **Service Compromise:** Server-Side Request Forgery (SSRF) attacks against EC2 instances provide an initial foothold for expansion (Sources and Reports, 2019; F5 Labs, 2025).

3. **Insider Threats:** Legitimate users abuse overpermissive policies to access unauthorized resources (Cappelli et al., 2012).

Identity Types and Chaining Detection. The `userIdentity.type` field reveals the nature of each caller and its position within attack chains:

Table 3: Key identity types and their role in attack chains.

Identity Type	Role in Attack Chains
`IAMUser`	The `IAMUser` type often represents the compromised starting point of attacks.
`AssumedRole`	The `AssumedRole` type is a primary indicator of ongoing role chaining.
`AWSService`	The `AWSService` type represents legitimate automation for filtering from analysis.
`FederatedUser`	The `FederatedUser` type is a corporate identity that can initiate chains.

The raw event distribution highlights the specific signal-versus-noise challenge that guides RoleSentry's design. Table 4 indicates that `AssumedRole` and `AWSService` make up 72.72% of all `AssumeRole` events, while `IAMUser` contributes only 0.11%. This overwhelming volume of automated and legitimate chained activity necessitates the filtering-first approach detailed in our methodology.

The background above establishes the IAM and CloudTrail context needed for detection. We now formalize the problem and present empirical observations that motivate our design.

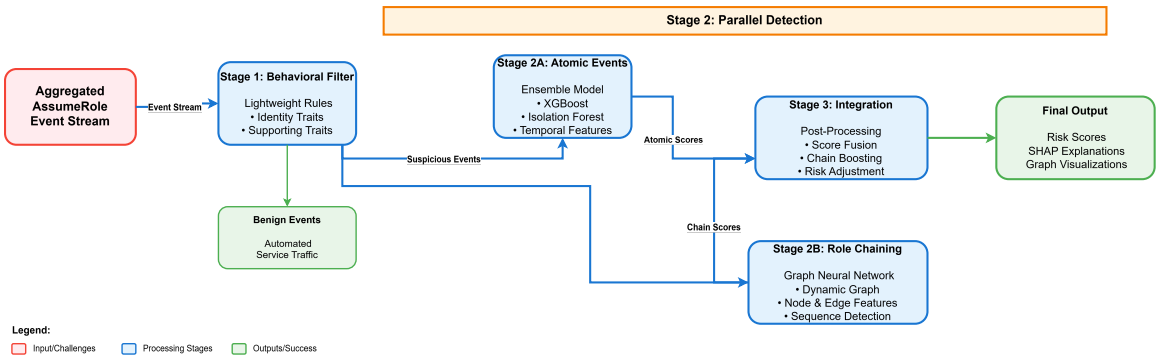

Figure 2: RoleSentry system architecture showing the three-stage detection pipeline that addresses core challenges through corresponding design goals. The system processes AssumeRole events through a behavioral filter (Stage 1), parallel atomic and chaining detection pipelines (Stage 2), and contextual integration (Stage 3) to produce explainable risk assessments.

3. RoleSentry: System Design and Methodology

Having established a critical background on AWS IAM, CloudTrail, and the threat of role chaining, this section details the architecture and methodology of RoleSentry. It formally defines the problems of detecting suspicious role assumptions, outlines the core challenges inherent in these tasks, and then provides a comprehensive description of RoleSentry's multi-stage design and its unique methodologies for robust threat detection.

3.1. Problem Definition and Core Challenges

Let $\mathcal{E} = \{e_1, e_2, \ldots, e_n\}$ represent a high-volume stream of `AssumeRole` events. Each event e_i contains detailed information as described in Section 2. For each event e_i, a true but unobservable label $l_i \in \{\text{benign}, \text{suspicious}\}$ exists. The domain presents an extreme class imbalance, a fundamental challenge (Wang et al., 2024). The set of benign, automated service-to-service events is orders of magnitude larger than all other activity, such that $|\mathcal{E}_{\text{benign, svc}}| \gg |\mathcal{E}_{\text{suspicious}}|$. RoleSentry addresses two distinct but related detection problems:

1. **Atomic Event Detection:** Determine whether a single AssumeRole event e_i is suspicious based on its individual characteristics and behavioral context.
2. **Role Chaining Detection:** Identify sequences of AssumeRole events that, while individually appearing benign, collectively form malicious multi-step attack patterns.

3.2. Empirical Motivation: Identity-Type Distribution and Design Implication

The raw event distribution highlights the signal-versus-noise challenge that guides RoleSentry's design. Table 4 shows that `AssumedRole` and `AWSService` account for 72.72% of all `AssumeRole` events, while `IAMUser` contributes only 0.11%. The volume of automated and legitimate chained activity motivates a filtering-first approach in Section 3.5.

Table 4: Distribution of AssumeRole events by `userIdentity.type`.

Identity Type	Event Count	% of Total
`AssumedRole`	203,891,565	37.71%
`AWSService`	189,313,583	35.01%
`AWSAccount`	146,859,972	27.16%
`IAMUser`	615,122	0.11%

Understanding these patterns enables RoleSentry to distinguish legitimate automation from attack sequences. The next subsections describe how the architecture leverages these insights for robust detection.

3.3. Core Challenges and Corresponding Design Goals

Any practical detection system must overcome three core challenges to prove effective in such a noisy environment. For each challenge, RoleSentry establishes a corresponding design goal that directly addresses the underlying problem, as visualized in Figure 2.

Challenge 1: Signal-versus-Noise & Design Goal: Filter First, Analyze Later

Challenge: Precisely distinguish suspicious events from overwhelming background noise of legitimate automation. Focus on identifying subtle indicators of role chaining, which may hide within high volumes of benign activity.

Design Goal: Explicitly filter predictable, benign service-to-service traffic with a lightweight mechanism. This prefiltering concentrates intensive deep analysis on a smaller, potentially higher-risk subset of events, including those that might

form part of a multihop role chain. *Motivating evidence:* See Table 4, where `AssumedRole`+`AWSService` compose 72.72% of events.

Challenge 2: Contextual Understanding & Design Goal: Model Behavior, Not Just Events

Challenge: Model an event's *behavioral context*, including its position within a potential role chain, rather than relying solely on statistical rarity. Understand the complex relationships and sequences between successive `AssumeRole` calls.

Design Goal: Employ a robust model that learns rich behavioral profiles from historical context. Identify deviations that point to individual suspicious events and to patterns that indicate parts of a larger role chain. Capture and reason about relational information inherent in sequential role assumptions.

Challenge 3: Operational Viability & Design Goal: Ensure Explainability

Challenge: Produce accurate, transparent, and actionable alerts that clarify whether an event is an isolated anomaly or part of a multistep attack. Reduce analyst fatigue and build trust in the system.

Design Goal: Accompany every high-risk alert with a clear, feature-based explanation. This transparency helps security analysts understand why the system flagged an event, particularly when the event relates to complex attack behaviors like role chaining. The clarity increases alert actionability and trust in the detection system.

3.4. RoleSentry Architecture Overview

RoleSentry designs its architecture as an integrated, multi-stage system that addresses both atomic event detection and role chaining detection through parallel processing pipelines. The architecture is specifically engineered to handle the high volume of cloud activity while efficiently identifying subtle threats. Figure 2 provides a comprehensive view of the system architecture, illustrating how each challenge maps to specific design goals and corresponding system components.

The system's flow is as follows:

1. **Stage 1: Behavioral Trait Filter.** Each incoming AssumeRole event first undergoes a lightweight, rule-based filter that rapidly inspects the event for empirically derived behavioral traits characteristic of benign automation.

2. **Stage 2: Parallel Detection Pipelines.** Events not confidently cleared by Stage 1 are processed through two parallel detection mechanisms:

   - **Atomic Event Pipeline:** An ensemble classifier assesses individual event anomalies using contextual and identity-based features.
   - **Role Chaining Pipeline:** A graph neural network analyzes temporal graphs of principal-to-role transitions to detect multi-step role chaining behavior.

3. **Stage 3: Contextual Post-Processing.** Raw risk outputs from both detection pipelines are fed into a final logic layer that applies domain-specific heuristics and contextual adjustments.

4. **Explainability Layer.** SHAP-based explanations and graph visualizations provide clear, feature-based explanations for all high-risk events.

3.5. Atomic AssumeRole Event Detection

This subsection details RoleSentry's approach to detecting suspicious individual AssumeRole events, addressing the first core detection problem. The components described here correspond to the left side of the parallel detection pipelines shown in Figure 2.

3.5.1. Stage 1: The Behavioral Trait Filter

The initial stage of RoleSentry employs a lightweight, rule-based filter designed to identify and triage rapidly events that exhibit strong characteristics of benign service-to-service automation. As illustrated in the first processing stage of Figure 2, this filter directly addresses Challenge 1 (Signal-versus-Noise) by implementing the '(Filter First, Analyze Later)' design goal.

Pre-filtering reduces the volume of data and makes computationally intensive downstream models more efficient, thus improving overall system efficiency and reducing false positives from normal operational churn. The filtering logic, $\text{IsBenignService}(e_i)$, receives formal definition as:

$$\text{IsBenignService}(e_i) := \text{IdentityTrait}(e_i) \wedge \left(\bigvee_{t \in \mathcal{T}_{\text{support}}} t(e_i) \right) \tag{1}$$

where 'IdentityTrait' represents a mandatory primary trait that must be present to identify service-to-service patterns, and $\mathcal{T}_{\text{support}}$ is a set of one or more supporting traits that further confirm benign automation.

- **Identity Traits** include events where the caller's `userIdentity.type` is explicitly 'AWSService' or 'AWSAccount'. Additionally, this trait captures cases where the assumed role ARN or user agent string contains explicit keywords, such as 'service-role,' 'automation,' or 'terraform.' These are strong indicators of automated, benign processes.

- **Supporting Traits** include indicators of automation behavior that reinforce an event's benign nature. Examples include a principal assuming a high number of distinct roles from a minimal, stable set of source IPs (indicating high-volume, low-cardinality automation) or a role being consistently assumed from a historically stable set of network locations.

Events not classified as benign by this filter proceed to the atomic event detection pipeline, as shown by the flow arrows in Figure 2.

3.5.2. Stage 2A: Hybrid Ensemble Model for Atomic Events

For events not filtered out by Stage 1, RoleSentry employs a hybrid ensemble model for deep anomaly detection based on point-in-time and aggregated temporal features. This component, labeled as '(Stage 2A: Atomic Events)' in Figure 2, directly implements Challenge 2's design goal of modeling behavior rather than just events.

The model combines the strengths of an Isolation Forest for outlier detection with an XGBoost classifier for supervised learning on labeled data (Rahman et al., 2024; Almutairi et al., 2024). The final score from this component, $R_{\text{atomic}}(e_i)$, represents a weighted combination of their outputs, with a weighting factor $\alpha = 0.7$ empirically determined to balance their contributions:

$$R_{\text{atomic}}(e_i) = \alpha \cdot P_{\text{XGB}}(y = 1|X_i) + (1 - \alpha) \cdot \mathcal{N}(S_{\text{IF}}(X_i)) \tag{2}$$

where X_i is the feature vector representing event e_i, $P_{\text{XGB}}(y = 1|X_i)$ represents the probability assigned by the XGBoost model that e_i is suspicious, $S_{\text{IF}}(X_i)$ is the anomaly score from the Isolation Forest, and $\mathcal{N}$ is a normalization function that scales the Isolation Forest score to be comparable with the XGBoost probability.

Advanced Feature Engineering To provide the ensemble model with a rich understanding of behavioral dynamics beyond simple counts, RoleSentry engineers a comprehensive set of temporal features. For each entity involved in an `AssumeRole` event (e.g., source IP, assumed role, calling principal), RoleSentry computes features over sliding time windows (e.g., 1-day, 7-day, 14-day, 30-day periods). These features include ratios, velocity and acceleration, and volatility metrics.

- **Ratios:** Ratios of short-term to long-term activity (e.g., `count_1d / count_7d`). A sudden increase in this ratio can indicate bursts of activity, which are often indicative of an attacker's initial credential use.
- **Velocity and Acceleration:** These metrics quantify the rate of change in activity levels and the rate of change of that velocity, respectively. They provide insights into principals' dynamic behavior patterns.
- **Volatility:** The standard deviation of activity, normalized by the mean, measures behavioral stability. Low stability (high volatility) can indicate an attacker's exploratory actions or other unpredictable behavior patterns.

3.6. Role Chaining Detection via Graph Neural Networks

This subsection details RoleSentry's approach to detecting multi-step role-chaining attacks, addressing the second core detection problem through graph-based analysis. The components described here correspond to the right side of the parallel detection pipelines in Figure 2, labeled as '(Stage 2B: Role Chaining.)'

3.6.1. Problem Formulation for Role Chaining

Role chaining detection requires modeling the sequential and relational nature of 'AssumeRole events. Unlike atomic event detection, which considers events in isolation, role-chaining

detection must capture the temporal dependencies and structural relationships between events. Let $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$ represent a set of potential role chains, where each chain c_j is a sequence of related AssumeRole events. The goal is to identify chains that exhibit malicious patterns while distinguishing them from legitimate operational sequences.

3.6.2. Stage 2B: Graph-Based Analysis for Chaining Detection

To explicitly model and detect complex, multi-hop role assumption chains, RoleSentry incorporates a Graph Neural Network (GNN) component. This component operates on a dynamic graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}_g)$ constructed from the stream of AssumeRole events, capturing the inherent relational structure of these activities. As shown in Figure 2, this pipeline runs in parallel with the atomic event detection to provide comprehensive coverage of both individual and sequential threat patterns.

Graph Construction The dynamic graph $\mathcal{G}$ builds in real-time as AssumeRole events process:

- **Nodes ($\mathcal{V}$):** Each node $v \in \mathcal{V}$ represents a distinct IAM entity involved in a role assumption. Specifically, this includes IAM users, individual assumed role sessions (uniquely identified by 'roleArn' and 'roleSessionName' combinations), and AWS services. Nodes initialize with feature vectors $h_v^{(0)}$ that encompass both static attributes (e.g., role permissions, associated AWS account ID) and aggregated behavioral features (e.g., historical activity counts and statistical metrics derived from the atomic detection pipeline).
- **Edges ($\mathcal{E}_g$):** A directed edge $(u, v) \in \mathcal{E}_g$ creates from node u to node v if principal u successfully assumes role v within a specified temporal window. This directed edge explicitly captures the AssumeRole relationship and its direction. Edges also enrich with features, such as the 'sourceIPAddress' from which the assumption occurred or the 'eventTime' difference between consecutive assumptions in a potential chain.

An adjacency matrix A represents the graph's connectivity, capturing AssumeRole relationships between entities over time.

GNN Architecture for Chaining Detection A Graph Neural Network (GNN) then applies to process this dynamic graph structure. GNNs are powerful deep learning models capable of learning embeddings that encode relational patterns, crucial for identifying role chaining. The GNN layers iteratively aggregate and transform information from a node's immediate and extended neighbors, effectively propagating context across the entire graph. For a node v, its embedding $h_v^{(k+1)}$ at layer $k+1$ computes by aggregating messages from its neighbors and transforming its embedding from the previous layer, commonly expressed as:

$$h_v^{(k+1)} = \sigma\left(W_1^{(k)} h_v^{(k)} + W_2^{(k)} \sum_{u \in \mathcal{N}(v)} \frac{1}{c_{vu}} h_u^{(k)}\right) \tag{3}$$

where $\mathcal{N}(v)$ are the direct neighbors of node v, $W_1^{(k)}$ and $W_2^{(k)}$ are learnable weight matrices for the current layer k, σ is a non-linear activation function (e.g., ReLU), and c_{vu} is a normalization constant.

For explicit chaining detection, the GNN can train for various tasks:

- **Node Classification:** Predict a risk score for each assumed role session node based on its structural position and features within detected chains.
- **Edge Classification:** Predict the likelihood that a specific AssumeRole event (represented as an edge) is part of a malicious chain.
- **Graph-level Anomaly Detection:** Identify entire subgraphs (sequences of interconnected role assumptions) that deviate significantly from learned benign patterns.

The output of the GNN component yields a risk score for each event e_i based on its relational context, denoted as $R_{\text{chain}}(e_i)$.

3.7. Integration and Final Risk Assessment

This subsection describes how RoleSentry integrates outputs from both detection pipelines to produce final risk assessments, corresponding to Stage 3 in Figure 2.

3.7.1. Stage 3: Contextual Post-Processing

Raw scores generated by the atomic event detection pipeline (R_{atomic}) and the role chaining detection pipeline (R_{chain}) are integrated through a contextual post-processing stage, as depicted in the '(Stage 3: Integration)' component of Figure 2. This stage incorporates domain-specific heuristics and contextual adjustments. The final risk score for an event e_i is computed as:

$$R(e_i) = \text{ContextualAdjust}\big(R_{\text{atomic}}(e_i), \quad R_{\text{chain}}(e_i), \\ \text{ChainContext}(e_i)\big) \tag{4}$$

where $\text{ChainContext}(e_i)$ captures additional contextual information about the event's position within detected chains.

A critical adjustment involves the explicit detection and subsequent escalation of risk for role-assumption chains. Suppose RoleSentry identifies an event as part of a multi-step chain, meaning the calling principal belongs to an 'AssumedRole' type and follows a prior assumed role within a suspicious time window. In that case, RoleSentry significantly boosts its risk score.

3.7.2. Explainability Integration

For all high-risk events identified by the final stage, RoleSentry integrates SHAP (Shapley Additive exPlanations) to provide clear, feature-based explanations for atomic events and graph visualizations to illustrate role-chaining patterns. This component addresses Challenge 3 (Operational Viability) by implementing the explainability design goal, as shown in the final output section of Figure 2. This dual-explainability approach ensures that security analysts can quickly understand whether an alert represents an isolated anomaly or part of a coordinated, multi-step attack.

4. Security-Focused Evaluation

To validate RoleSentry's effectiveness and operational viability, we conduct a comprehensive evaluation of its atomic and role-chaining detection capabilities. The evaluation uses a substantial real-world dataset and includes a direct comparative analysis against Amazon GuardDuty alerts. GuardDuty is a threat detection offering by AWS for its cloud customers(Amazon Web Services, 2024a). We focus on demonstrating RoleSentry's ability to reduce false positives while accurately detecting high-fidelity threats, including complex role-chaining scenarios.

4.1. Dataset and Ground Truth

Dataset Preparation and Feature Engineering. We evaluate RoleSentry's performance using a 30-day dataset from a large enterprise environment, comprising 540 million raw `AssumeRole` events. The raw data feeds into two separate, parallel processing infrastructures. First, for atomic event analysis, our automated feature engineering pipeline transforms the raw logs. For each event, this pipeline computes a rich set of behavioral features by aggregating activity over multiple time windows (1-day, 7-day, 14-day, and 30-day). This process yields our primary analytical dataset of 4 million feature-rich rows. Second, for role-chaining analysis, a separate infrastructure processes the raw event stream to construct a graph of principal-to-role relationships. From this graph, we explicitly identify 962 role chaining events, forming the dataset for our GNN-based evaluation.

Ground Truth Threat Scenarios. The ground truth for our evaluation consists of 75 **known attacks** present within the dataset. These known attacks cover a range of Tactics, Techniques, and Procedures (TTPs), including credential compromise, server-side request forgery (SSRF) for privilege escalation, and insider threat patterns. This approach ensures our ground truth reflects the complexity and subtlety of real attack behavior, providing a robust benchmark for evaluating detection performance against a dataset dominated by benign automated activity (as detailed in Table 4).

4.2. Experimental Design and Validation Protocol

To ensure a rigorous evaluation and prevent data snooping, we follow established best practices for temporal data analysis (Arp et al., 2022; Wang et al., 2024). We divide the data chronologically into training (70%), validation (15%), and test (15%) sets, with 24-hour buffer gaps between sets to prevent temporal data leakage. We compute all feature engineering statistics, normalization parameters, and categorical encodings exclusively on the training data before applying them to the validation and test sets. The approach represents a realistic deployment in which the model must classify future events using only historical data, ensuring our reported performance metrics reflect genuine operational capabilities.

4.3. Atomic Event Detection Performance

This evaluation assesses RoleSentry's ability to detect suspicious individual `AssumeRole` events using its two-stage atomic detection pipeline.

Prefiltering Effectiveness and Volume Reduction. The Stage 1 Behavioral Trait Filter is crucial for managing the dataset's scale (see section 3.5.1). The filter successfully identifies

and suppresses a substantial volume of benign automation events, which is vital for reducing analyst workload and focusing computational resources.

Figure 3 illustrates the behavioral separation the filter achieves, where events with a high service-account likelihood consistently receive low risk scores. Significantly, the filter produces no conflicting classifications; no event is flagged as both likely automation and high risk, demonstrating its reliability and eliminating ambiguous alerts.

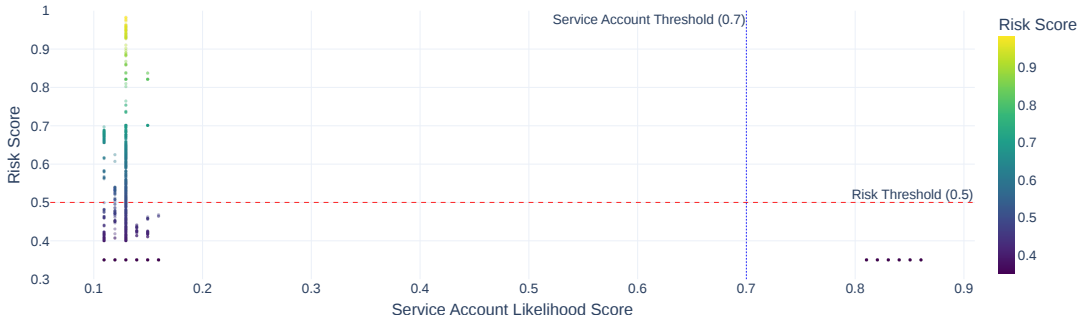

Figure 3: Service account detection performance: risk score versus service-account likelihood score. The inverse relationship separates automation (high service scores, low risk) from suspicious activity (low service scores, high risk). Data clustering in the lower-left quadrant confirms the behavioral distinction between legitimate automation and malicious activity.

Risk Scoring and Anomaly Discrimination. For events that pass the initial filter, RoleSentry's hybrid ensemble model assigns a risk score to identify suspicious activity. The model effectively distinguishes the 75 confirmed malicious scenarios from the millions of normal, non-automated user events in the dataset. Figure 4 shows the model's strong discrimination power, particularly in the high-risk tail above the 95th percentile. The weak negative correlation (-0.046) between the service likelihood and risk scores further validates our hypothesis that the model learns fundamentally different behavioral patterns for legitimate automation versus suspicious user activity.

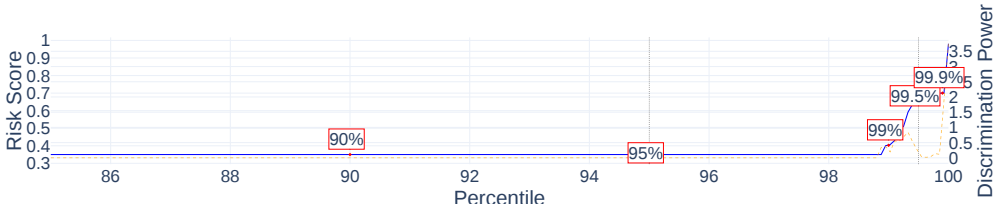

Figure 4: Risk score distribution by percentile (blue line) and model discrimination power (orange dashed line). Red diamonds mark key thresholds, while gray dotted lines indicate decision boundaries at 95th and 99.5th percentiles. The model shows minimal discrimination in the 0-90th percentile range but strong separation capability in the high-risk tail above the 95th percentile.

4.4. Role Chaining Detection Performance

We evaluate the Graph Neural Network (GNN) component on the 962 role chains extracted from the dataset to assess its ability to distinguish malicious chaining patterns from legitimate operational workflows. The subsection first describes observed chaining behavior in a typical environment, including hop prevalence and event durations. It then presents RoleSentry's detection results for multi-hop `AssumeRole` attacks.

Observed Chain Characteristics. Most chains in the dataset are short in both length and duration. Over 90% contain exactly two hops (Figure 6). In terms of timing, 84.1% complete within one minute (Figure 5). These observations characterize typical chaining in our data. Longer chains or extended durations are uncommon and provide helpful cues for the GNN to prioritize for review.

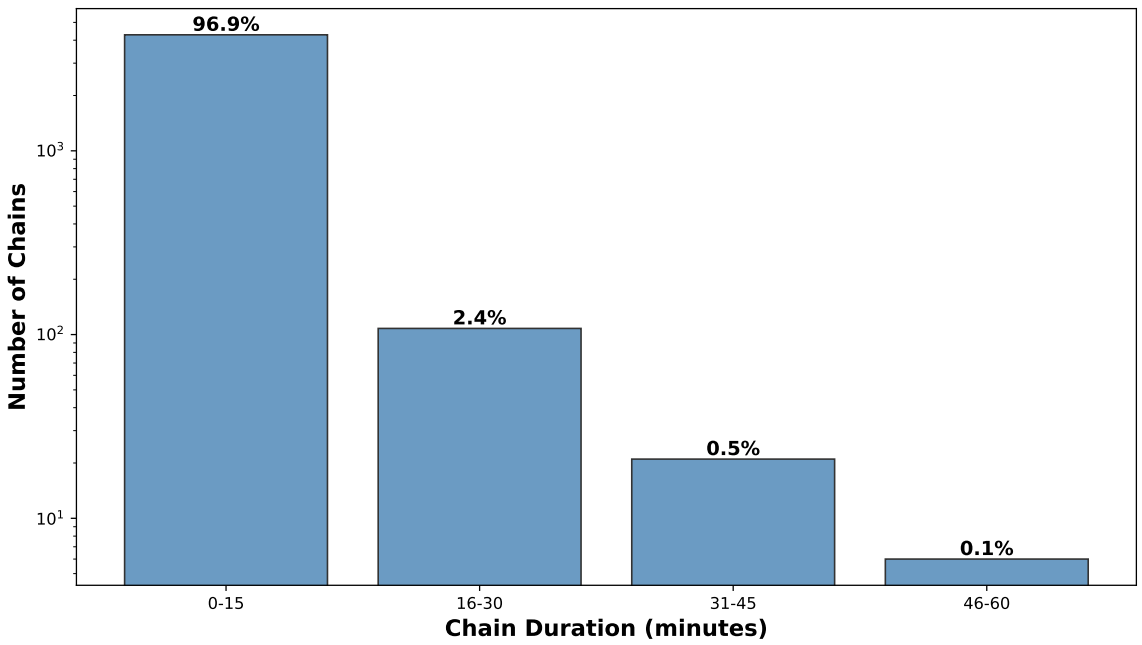

Figure 5: End-to-end role chain durations in minutes. Duration equals elapsed time from the first to the last `AssumeRole` hop. 96.9% finish within 15 minutes;

AWS Constraints and Practical Implications. When you assume a role with long-term credentials, you can request up to 43,200 seconds (12 hours), subject to the role's configured maximum session duration Amazon Web Services (2025d). When you assume a role using credentials from a previously assumed role (role chaining), AWS caps the session at 3,600 seconds (1 hour) regardless of the role's maximum. Any role-chained request with `DurationSeconds` greater than 3,600 fails (Amazon Web Services, 2024e). In our dataset, nearly all `AssumeRole` sessions ended by the 50th minute. A configuration review showed that the maximum session duration for all roles was 60 minutes. Because session credentials can be stolen, we recommend keeping role session durations at 60 minutes or lower to reduce the window of opportunity for misuse.

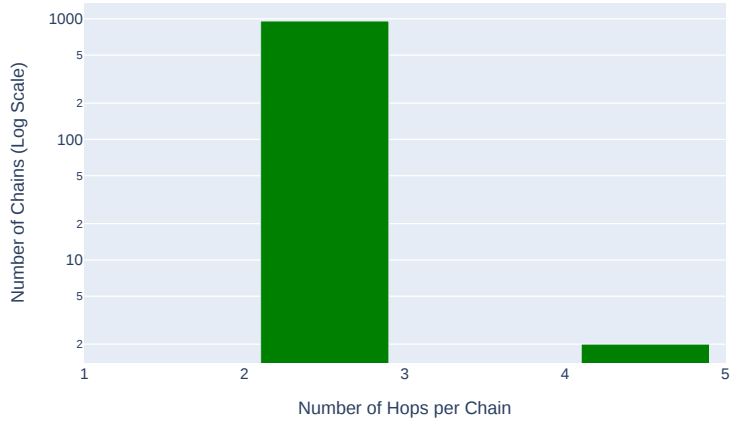

Figure 6: Chain Length Distribution (Hops per Chain) showing the predominance of short chains in typical enterprise environments.

Detection Results. The GNN flags 7 of the 962 chains as high risk, filtering out 99.3%. Manual review confirms 6 of these 7 alerts as true positives, yielding 85.7% precision. The remaining alert was not a true positive; a developer was interacting with AWS via unsanctioned, ad hoc workflows. Although not malicious, the security team advised the developer to move to approved, paved engineering paths. The graph model captures structurally atypical patterns that emerge from node and edge context over time, including rare principal transitions, unusual source IP ranges, uncommon user agents, and cross-account paths not observed in historical chains. More importantly, the GNN surfaces semantically suspicious patterns that appear ordinary by length but are structurally anomalous. For example, a two-hop chain initiated from an unfamiliar network range with an unusual user agent and an eight-minute gap between hops. Graph context over nodes and edges, including principal transitions, source networks, user agents, and cross-account paths, enables RoleSentry to flag cases that simple timing rules or per-event heuristics miss.

4.5. Comparative Analysis Against Amazon GuardDuty

To benchmark RoleSentry's operational value, we compare its findings with Amazon GuardDuty using our ground-truth dataset. The analysis reveals a critical performance gap. RoleSentry achieves a 98% reduction in false positive alerts compared to GuardDuty, which frequently flags benign operational changes (such as the deployment of a new CI/CD pipeline) as anomalous. More critically, GuardDuty fails to detect any of the 25 most sophisticated, multi-step attacks present. The 25 events a subset of the 75 events. These missed detections include compromised IAM credentials used from attacker-controlled infrastructure, SSRF

Table 5: Detailed comparison of RoleSentry and Amazon GuardDuty on nuanced, real-world event scenarios.

Event Scenario Description	**GuardDuty Finding Type**	**Analyst Verdict**	**GuardDuty Alert**	**RoleSentry Alert**
A new CI/CD pipeline service is deployed and assumes multiple roles for the first time.	`IAMUser/AnomalousBehavior`	Benign	**Yes**	No
Total Benign Operational Events Flagged by GuardDuty	*(50 events)*		*50*	*1 (98.0% reduction)*
A compromised IAM user key was used from an attacker-controlled server in a new country.	(No Finding)	Suspicious	No	**Yes**
SSRF on an EC2 instance is used to assume a high-privilege role that the instance has never used.	(No Finding)	Suspicious	No	**Yes**
An insider logs in from their home network at 3 AM, assumes their role, and accesses sensitive data through a multi-step role chain.	(No Finding)	Suspicious	No	**Yes**
Total Confirmed Suspicious Events	*(25 events)*		*0*	*25*

exploitation for privilege escalation, and insider threats leveraging role chaining. Table 5 provides a detailed side-by-side comparison of specific scenarios that highlight these performance differences. The results confirm RoleSentry's superior ability to differentiate benign operational events from genuine threats in a noisy, real-world cloud environment.

5. Threats to Validity

While RoleSentry demonstrates strong performance and operational benefits, several limitations merit consideration for contextualizing findings and guiding future research:

External Validity: The evaluation data was collected from a single enterprise environment. While this dataset is extensive and representative of large-scale AWS operations, findings may not generalize perfectly to all AWS environments, which vary significantly in operational postures, security configurations, and usage patterns. Specific behavioral

traits identified might require re-calibration for organizations with different cloud footprints or architectural designs. Furthermore, observed role chaining patterns could vary across different organizations.

Construct Validity: Ground truth for both benign and suspicious activities relies on a limited number of confirmed security incidents and synthetic attacks. While these synthetic attacks were meticulously crafted based on known Tactics, Techniques, and Procedures (TTPs), they may not cover all novel or highly sophisticated attack techniques, especially those involving new forms of evasion or complex role chaining methods. The GNN component's effectiveness intrinsically depends on the completeness and accuracy of graph representations derived from `AssumeRole` events.

Implementation-Specific Threats: Advanced, persistent attackers could potentially learn the system's detection traits and behavioral models, allowing them to craft role chaining patterns that mimic benign behavior and evade detection. Additionally, RoleSentry's evaluation is based on a static dataset. Live, continuously evolving environments would require periodic model retraining to address concept drift and adapt to evolving attacker techniques, particularly concerning dynamic graph structures and GNN model updates.

Explainability Limitations: Our framework relies on SHAP for local explanations. We acknowledge the method's inherent limitations, as recent research demonstrates that Shapley values can produce inconsistent or misleading explanations, particularly in the presence of correlated features (Huang and Marques-Silva, 2024). While the explanations proved helpful for analyst interpretation during our evaluation, future work could explore complementary explainability techniques to provide a more robust understanding of model decisions.

Despite these limitations, RoleSentry provides a robust foundation for future advancements in cloud security threat detection.

6. Related Work

Multi-Step IAM Attack Detection. Prior work has explored formal verification approaches for cloud security analysis. Prizmant et al. (Prizmant et al., 2022) present model-checking techniques for detecting multi-step IAM attacks in AWS environments, with a focus on pre-deployment policy verification. However, these formal approaches analyze static configurations rather than runtime behavioral patterns, which limits their applicability to operational threat detection in high-volume production environments where distinguishing legitimate automation from malicious activity is crucial.

Graph-Based Security Analysis. Graph neural networks have shown promise for cybersecurity applications, with recent surveys (Wang et al., 2024) examining their effectiveness in intrusion detection systems. Wang et al. explore GNN applications for threat intelligence analysis (Li et al., 2024), while Zhang et al. (Zhang et al., 2024) address privacy considerations in graph-based security models. However, the application of GNNs to cloud IAM role chaining represents a previously unexplored area requiring specialized temporal graph modeling approaches.

Explainable AI in Cybersecurity. Recent research has demonstrated the effectiveness of explainable AI techniques in security applications. Rahman et al. (Rahman et al., 2024) and Almutairi et al. (Almutairi et al., 2024) show improved performance using SHAP-based

explanations in intrusion detection, while recent work (Almutairi et al., 2024) applies XAI to encrypted malware detection. These approaches validate the importance of transparency in security decision-making, although none specifically address cloud IAM analysis.

Cloud Security and Behavioral Analysis. AWS security research has documented various role chaining attack patterns (Elastic, 2024; Levine and Sonntag, 2021; CrowdStrike, 2024b) and exploitation techniques (Labs, 2021; Security, 2020; Splunk Threat Research Team, 2021). Commercial solutions like Amazon GuardDuty (Amazon Web Services, 2024a) attempt to address these challenges but suffer from high false positive rates (Skyhawk Security, 2024; CloudOptimo, 2025b). Recent work on behavioral analysis for cloud environments (Salman et al., 2017) demonstrates the potential of AI-powered approaches for detecting token exchange anomalies, highlighting the importance of behavioral context in cloud security.

Privacy-Preserving Security Analytics. Emerging research explores federated learning for collaborative threat detection (Han et al., 2024; Hendaoui et al., 2025), enabling organizations to share threat intelligence while preserving sensitive data. While promising for future multi-organization scenarios, current approaches focus primarily on IoT and network security rather than cloud IAM analysis.

Positioning of RoleSentry. RoleSentry bridges the gap between theoretical security analysis and operational threat detection by combining behavioral filtering, ensemble anomaly detection, and graph neural networks specifically for AWS role-chaining detection. Our work is the first to apply GNNs to IAM role chain analysis. It addresses the critical operational challenge of distinguishing malicious activity from legitimate automation in high-volume cloud environments, a problem that existing formal verification and commercial solutions cannot effectively solve.

7. Conclusion

The challenge of detecting suspicious AWS role assumptions is fundamentally a signal-versus-noise problem, a complexity significantly compounded by sophisticated multi-step attacks like role chaining. RoleSentry fundamentally shifts the focus from merely finding statistical outliers to robustly modeling and effectively filtering legitimate service behavior. The paper introduced this novel detection framework. Its integrated, multi-stage architecture (combining a behavioral trait filter for efficient triage, an explainable ensemble model for deep anomaly detection, and contextual post-processing explicitly augmented by graph-based analysis (e.g., GNNs)) achieves high precision and exceptional operational viability. Crucially, RoleSentry's sophisticated detection capabilities, particularly its nuanced approach to role chaining, directly address a critical attacker technique that often bypasses traditional security tools. When benchmarked against Amazon GuardDuty, RoleSentry significantly reduces false positives stemming from benign operational churn and successfully detects high-fidelity threats, including subtle role chaining indicators, missed by the native service. Future work focuses on expanding the framework's evaluation across more diverse cloud environments and developing methodologies for continuous model adaptation to effectively counter evolving threat landscapes and new attack patterns.

References

S. Almutairi, A. M. AlRassas, and M. A. A. Al-qaness. Consensus hybrid ensemble machine learning for intrusion detection with explainable ai. *ScienceDirect*, 2024. Accessed: 2025-06-26.

Amazon Web Services. Amazon GuardDuty user guide. https://docs.aws.amazon.com/guardduty/latest/ug/what-is-guardduty.html, 2024a. Accessed: 2025-06-14.

Amazon Web Services. Enhancing multi-account activity monitoring with event-driven architectures. *AWS Compute Blog*, 2024b. Accessed: 2025-06-22.

Amazon Web Services. Suppression rules in GuardDuty. https://docs.aws.amazon.com/guardduty/latest/ug/findings_suppression-rule.html, 2024c. Accessed: 2025-06-22.

Amazon Web Services. Investigating lateral movements with Amazon Detective investigation and Security Lake integration. *AWS Security Blog*, May 2024d. Accessed: 2025-06-22.

Amazon Web Services. Assumerole - AWS Security Token Service. https://docs.aws.amazon.com/STS/latest/APIReference/API_AssumeRole.html, 2024e. Accessed: 2025-06-22.

Amazon Web Services. What is AWS CloudTrail? https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-user-guide.html, 2025a. Accessed: 2025-06-22.

Amazon Web Services. Security best practices in IAM. https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html, 2025b. Accessed: 2025-06-22.

Amazon Web Services. Mapping AWS security services to MITRE frameworks for threat detection and mitigation. https://aws.amazon.com/blogs/security/mapping-aws-security-services-to-mitre-frameworks-for-threat-detection-and-mitigation/, 2025c. Accessed: 2025-06-22.

Amazon Web Services. Iam roles. https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html#id_roles_terms-and-concepts, 2025d. Accessed: 2025-10-04.

Daniel Arp, Erwin Quiring, Feargus Pendlebury, Alexander Warnecke, Fabio Pierazzi, Christian Wressnegger, Lorenzo Cavallaro, and Konrad Rieck. Dos and don'ts of machine learning in computer security. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 3971–3988, 2022.

AWS Community. GuardDuty false positive rates. https://repost.aws/questions/QUYWBwmzV4SF-WOg5j3r5eFg/guardduty-false-positive-rates, 2024. AWS re:Post Community Discussion, Accessed: 2025-06-22.

D. M. Cappelli, A. P. Moore, and R. F. Trzeciak. *The CERT Guide to Insider Threats: How to Prevent, Detect, and Respond to Information Risk*. Addison-Wesley Professional, 2012.

CloudOptimo. How to leverage AWS CloudTrail to stay compliant and secure in the cloud? https://www.cloudoptimo.com/blog/how-to-leverage-aws-cloudtrail-to-stay-compliant-and-secure-in-the-cloud/, 2025a. Accessed: 2025-06-22.

CloudOptimo. AWS GuardDuty: Advanced threat detection for cloud security. https://www.cloudoptimo.com/blog/aws-guardduty-advanced-threat-detection-for-cloud-security/, 2025b. Accessed: 2025-06-22.

CrowdStrike. Cloud logs: The unsung heroes of detection and response. *CrowdStrike Blog*, 2024a. Accessed: 2025-06-22.

CrowdStrike. AWS lateral movement attack demo. https://www.crowdstrike.com/en-us/resources/videos/aws-lateral-movement-attack/, 2024b. Accessed: 2025-06-22.

Datadog. Best practices for monitoring AWS CloudTrail logs. *Datadog Security Blog*, 2024. Accessed: 2025-06-22.

John Dinsdale and CRN Staff. Cloud Market Share Q1 2025: AWS Dips, Microsoft And Google Show Growth. *CRN*, May 2025. Based on data from Synergy Research Group.

Edge Delta. Amazon CloudTrail challenges & solutions for secure logging. https://edgedelta.com/company/blog/amazon-cloudtrail-challenges, 2025. Accessed: 2025-06-22.

Elastic. AWS STS role chaining detection rules overview. https://www.elastic.co/guide/en/security/current/aws-sts-role-chaining.html, 2024. Accessed: 2025-06-22.

Security Experts. Addressing security analyst fatigue in the SOC. https://www.brighttalk.com/webcast/288/224207, 2024. Accessed: 2025-06-22.

F5 Labs. Campaign targets Amazon EC2 instance metadata via SSRF. *F5 Labs Articles*, 2025. Accessed: 2025-06-22.

Qingdi Han, Siqi Lu, Wenhao Wang, Haipeng Qu, Jingsheng Li, and Yang Gao. Privacy preserving and secure robust federated learning: A survey. *Concurrency and Computation: Practice and Experience*, 36(13):e8084, 2024.

Fatma Hendaoui, Rahma Meddeb, Lamia Trabelsi, Ahlem Ferchichi, and Rawia Ahmed. Fladen: Federated learning for anomaly detection in iot networks. *Computers & Security*, 155:104446, 2025.

Xuanxiang Huang and Joao Marques-Silva. On the failings of shapley values for explainability. *Int. J. Approx. Reasoning*, 171(C), August 2024. ISSN 0888-613X. doi: 10.1016/j.ijar.2023.109112. URL https://doi.org/10.1016/j.ijar.2023.109112.

Invictus Security. Dangerdev: Uncovering a sophisticated aws attack pattern. *Invictus Security Report*, 2024. Accessed: 2024-06-21.

Datadog Security Labs. Security vulnerabilities in aws amplify and role assumption. *Datadog Security Blog/Report*, 2023. Accessed: 2024-06-21.

Rhino Security Labs. Assume the worst: Abusing aws assumerole for privilege escalation. *Rhino Security Labs Blog/Report*, 2021. Accessed: 2024-06-21.

M. Levine and L. Sonntag. Cloud threat hunting: Attack & investigation series- lateral movement – under the radar. *Check Point Security Blog*, January 2021. Accessed: 2025-06-22.

Langsha Li, Feng Qiang, and Li Ma. Advancing cybersecurity: Graph neural networks in threat intelligence knowledge graphs. In *Proceedings of the International Conference on Algorithms, Software Engineering, and Network Security*, pages 737–741, 2024.

MITRE Corporation. Enterprise ATT&CK technique T1552: Unsecured credentials. https://attack.mitre.org/techniques/T1552/, 2024a. Accessed: 2025-06-14.

MITRE Corporation. Account manipulation: Additional cloud roles, sub-technique T1098.003 - enterprise. https://attack.mitre.org/techniques/T1098/003/, 2024b. Accessed: 2025-06-22.

MITRE Corporation. MITRE ATT&CK for AWS: Understanding tactics, detection, and mitigation. https://www.stream.security/post/mitre-attck-for-aws-understanding-tactics-detection-and-mitigation, 2024c. Accessed: 2025-06-22.

Netskope Research Team. MITRE att&ck view: Securing AWS temporary tokens. *Netskope Security Blog*, June 2021. Accessed: 2025-06-22.

Orca Security. What is lateral movement in cybersecurity? https://orca.security/glossary/lateral-movement/, 2024. Accessed: 2025-06-22.

Orca Security Research Team. The anatomy of an IAM cyber attack on AWS. *Orca Security Blog*, April 2023. Accessed: 2025-06-22.

D. Prizmant, S. Ravid, and A. Segev. Detecting multi-step IAM attacks in AWS environments via model checking. In *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 2022.

M. A. Rahman, S. Ahmed, and R. Khan. Hybrid bagging and boosting with shap based feature selection for enhanced predictive modeling in intrusion detection systems. *Scientific Reports*, 14, 2024. Accessed: 2025-06-26.

Tara Salman, Deval Bhamare, Aiman Erbad, Raj Jain, and Mohammed Samaka. Machine learning for anomaly detection and categorization in multi-cloud environments. In *2017 IEEE 4th international conference on cyber security and cloud computing (CSCloud)*, pages 97–103. IEEE, 2017.

Praetorian Security. Understanding and exploiting aws assumerole. *Praetorian Security Blog/Report*, 2020. Accessed: 2024-06-21.

SentinelOne. Securing AWS Lambda — how misconfigurations can lead to lateral movement. https://www.sentinelone.com/blog/lateral-movement-in-aws-lambda-environments/, 2024. Accessed: 2025-06-22.

I. Shevrin and O. Margalit. Detecting multi-step IAM attacks in AWS environments via model checking. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX, 2023.

Skyhawk Security. 10 pros and cons of AWS GuardDuty. https://skyhawk.security/pros-and-cons-of-aws-guardduty/, 2024. Accessed: 2025-06-22.

Various News Sources and Security Reports. The capital one data breach: Leveraging ssrf to data exfiltration. *Security Incident Analysis*, 2019. A widely reported incident in cloud security history.

Spacelift. AWS IAM roles - everything you need to know & examples. https://spacelift.io/blog/aws-iam-roles, 2024. Accessed: 2025-06-22.

Splunk Threat Research Team. Detecting AWS IAM privilege escalation. *Splunk Security Blog*, March 2021. Accessed: 2025-06-22.

Sysdig. Aws cloudtrail vs cloudwatch: Log differences. https://sysdig.com/blog/cloud-log-management-cloudwatch-vs-cloudtrail/, 2022. Accessed: 2025-06-21.

Sysdig Research Team. Exploiting IAM security misconfigurations. *Sysdig Security Blog*, May 2023. Accessed: 2025-06-22.

Shahroz Tariq, Mohan Baruwal Chhetri, Surya Nepal, and Cecile Paris. Alert fatigue in security operations centres: Research challenges and opportunities. *ACM Computing Surveys*, 2024. Accessed: 2025-06-22.

Horizon3.ai Research Team. Common aws misconfigurations enabling lateral movement. *Horizon3.ai Blog/Report*, 2022. Accessed: 2024-06-21.

Sysdig Research Team. Scarleteel: Lateral movement through containerized applications. *Sysdig Security Blog/Report*, 2023. Accessed: 2024-06-21.

Unit 42, Palo Alto Networks. Navigating the cloud: Exploring lateral movement techniques. *Unit 42 Security Research*, June 2024. Accessed: 2025-06-22.

Virtu. The pros and cons of GuardDuty. https://virtu.net/the-pros-and-cons-of-guardduty/, 2023. Accessed: 2025-06-22.

L. Wang, Y. Zhang, and Q. Chen. A survey on graph neural networks for intrusion detection systems: Methods, trends and challenges. *Computers and Security*, 141, 2024. Accessed: 2025-06-26.

H. Zhang, Y. Liu, X. Wang, and S. Chen. A survey on privacy in graph neural networks: Attacks, preservation, and applications. *IEEE Transactions on Knowledge and Data Engineering*, 36(12), 2024. Accessed: 2025-06-26.