Cairo University
Faculty of Computing and Artificial Intelligence

SIT495 – AI & ML in Cybersecurity
Course Project - Fall 2026

AI Cloud Security Investigator

Prepared by

Ganna Mohamed Abdelrahman	20236024
Habiba Khaled Ibrahim	20235008
Mariam Ahmed Mohamed	20235040
MennaTullah Hisham Hussein	20236101
Muhammed Hossam Sherif	20235028

Under the supervision of
Dr. Mohamed Ahmed Wahby

18 November 2026

1. Introduction

Moving workloads to the cloud has changed the way security investigations are carried out. In traditional networks, investigators can examine a device directly, create a disk image, or capture network traffic. However, in cloud environments such as Amazon Web Services (AWS), most activities happen through API calls, and the evidence of these activities is stored across different services [1]. For example, AWS CloudTrail records who performed an API call and when it happened, IAM provides information about users, roles, and permissions, S3 access logs show how data was accessed, and VPC Flow Logs provide details about network traffic [2]. Container and Kubernetes audit logs can also provide useful evidence. The challenge is that each service stores its logs in a different format, uses its own timestamps, and identifies users and resources differently. As a result, looking at one source alone is rarely enough to understand what actually happened during an attack.

This creates a major challenge in cloud security investigations: reconstructing an attack from evidence scattered across multiple cloud services is a time-consuming and complicated process [1], [3]. Investigators need to bring these logs together, identify connections between users, resources, and network activity, and distinguish suspicious actions from normal administrative activities. They also need to arrange all events in the correct order to understand how the attack developed.

A typical cloud attack may involve several stages, starting with an attacker gaining access through stolen credentials, followed by privilege escalation, persistence, lateral movement, unauthorized data access, and potentially data exfiltration [4]. Understanding how these actions are connected is essential for building an accurate incident timeline. This timeline should also be linked to the MITRE ATT&CK Cloud matrix [5] to help security teams identify the techniques used by the attacker and decide which actions are needed to contain the incident. However, performing all these steps manually, especially when dealing with thousands of log events, takes considerable time and increases the possibility of human error.

To address this challenge, our project, the AI Cloud Security Investigator, aims to automate the investigation process. The system collects logs from different AWS services, converts them into a common format, and analyzes them to identify connections between related events and build possible attack timelines. It then maps suspicious activities to relevant MITRE ATT&CK techniques and uses an AI layer to generate an understandable investigation report, including suggested containment actions.

Throughout this project, we explore different AI approaches that could support these tasks, includingBayesian decision theory, XGBoost and retrieval-augmented model (RAG). We compare these approaches to understand how each can contribute to different stages of the investigation process.

And to complement the automated investigation process, our project uses classification as an additional source of insight. By assigning each security event to an attack type, or to normal activity, classification gives analysts an immediate, measurable view of what is happening, alongside the timelines and reports that the system generates.

We chose classification after reviewing the learning techniques covered in the course: classification, regression, Probably Approximately Correct (PAC) learning, Bayesian decision theory, parametric learning, and unsupervised crisp clustering. Of these, classification best fits our problem, because attack types are discrete categories, labeled examples are available, and its results can be evaluated objectively. The other techniques are better suited to supporting roles, such as risk scoring, decision rules, or grouping related events

Why We Chose This Area

We chose this topic for several reasons. First, cloud security has become an important part of modern cybersecurity, making effective incident investigation increasingly necessary. Security teams often deal with evidence spread across different services, which makes connecting events and understanding the full attack difficult.

Second, the topic is closely related to what we study in this course. It provides an opportunity to explore how machine learning can help analyze large volumes of security logs while also understanding the limitations of AI, such as false positives and incorrect or misleading explanations [3], [6]. These limitations are particularly important in digital forensics, where investigation results need to be accurate and supported by evidence.

Finally, with the continuous development of technology and the rapid evolution of AI and machine learning, the amount and complexity of data generated by modern systems are constantly increasing. As a result, it is becoming more difficult for human analysts to identify every important detail, recognize hidden patterns, and connect events across large amounts of data through manual investigation alone. Our project aims to support analysts by helping them identify relevant evidence, uncover connections that might otherwise go unnoticed, and make the investigation process faster and more efficient. Rather than replacing human investigators, the system is designed to assist them in making more informed decisions based on a clearer and more complete picture of the incident.

Importance of the Project

Automating parts of the investigation process can help security teams understand incidents faster and reduce the time between detecting an attack and taking action to contain it. Instead of manually reviewing large amounts of raw logs, investigators can work with a structured timeline that explains what happened, how different events are connected, and which actions should be considered next.

However, speed alone is not enough. Since this system is intended for security investigations, its results must also be reliable and explainable. For this reason, evaluating the system is an important part of our project. We plan to measure how accurately the generated timelines match the known steps of simulated attacks and document cases where the AI produces incorrect results or fails to identify important activities.

2. Market Products Survey

2.1 AWS Security Incident Response

Product Overview

AWS Security Incident Response is a managed incident response service by Amazon Web Services, generally available since December 2024 [7]. In November 2025 AWS added an AI investigative agent to it [8]. Pricing is per finding ingested, with the first 10,000 findings per month free [9].

Main Purpose

To help organizations prepare for, respond to and recover from security incidents in AWS by combining automated triage, AI-assisted investigation and 24/7 support from AWS security engineers [10].

Main Features

- Automatic monitoring and triage of security findings [10].
- Case management with notification of the customer's response team [10].
- Containment actions run on the customer's behalf, with permission [10].
- AI investigation that produces a summary, timeline and recommendations [11].
- 24/7 access to AWS incident response engineers [10].

Architecture and Working Mechanism

- Enabled organization-wide from the AWS Organizations management account or a delegated administrator [12].
- Findings arrive from GuardDuty and third-party tools through Security Hub and are triaged using customer context such as known IPs and IAM entities [12].
- Findings that need attention open a case and the response team is notified [10].
- On AWS-supported cases the AI agent starts automatically, gathers evidence through a read-only service-linked role, and posts its summary to the case [11].

Data Sources

- Findings: Amazon GuardDuty and supported third-party tools via AWS Security Hub [12].
- AI investigation: AWS CloudTrail, IAM, Amazon EC2 instance APIs and AWS Cost Explorer [11].

AI/ML and Automation Capabilities

- Accepts a natural-language description of the incident; asks optional clarifying questions and starts on its own after 10 minutes [11].
- Correlates evidence across sources and builds a timeline of events [11].
- Writes an investigation summary with recommendations, typically within minutes [11].

- Agent actions are logged in CloudTrail; AWS states customer data is not used for model training [11].

Strengths

- Native to AWS: no connectors or log shipping needed.
- Human experts available 24/7 [10].
- Covers triage, investigation and containment in one service.
- Free for small workloads (first 10,000 findings per month) [9].

Limitations

Based on AWS's public documentation; a capability not described there may still exist.

- The AI produces one summary per case; the documentation does not describe analysts asking follow-up questions about the logs [11].
- Users must verify AI findings themselves, but the documentation does not describe showing the queries or the events behind each statement [11].
- The AI agent needs AWS Organizations and works only on AWS-supported cases [11].
- Closed model and prompts, with no published accuracy figures [11].

Relevance to Our Project

- Closest commercial alternative: same cloud (AWS), same data (CloudTrail) and same goal (AI investigation).
- Our project fills the analyst-side gap: plain-English follow-up questions translated to SQL, the exact query shown with every answer, and every report statement linked to its CloudTrail events.
- Ours runs on any CloudTrail data (a single account, a sandbox or exported logs) and its accuracy can be measured on public datasets.
- It complements AWS Security Incident Response rather than replacing its human responders and containment.

2.2 Amazon Detective (AWS)

Product Overview

Amazon Detective automatically collects log data from AWS resources and uses machine learning, statistical analysis and graph theory to build a linked data set for security investigations [13]. It is enabled in a few clicks in the AWS console or through an API, with no software to deploy and no log feeds to enable. It is a regional service, supports multiple accounts under one administrator account, offers a 30-day free trial, and is priced per GB of data ingested per account, region and month [13], [14]. It keeps up to one year of aggregated data [15].

Main Purpose

Detective helps analysts analyze, investigate and quickly find the root cause of potential security issues or suspicious activity [13]. It complements other AWS services instead of replacing them: GuardDuty detects threats, Security Hub aggregates findings, and Detective investigates them.

Main Features

- Behavior graph: a unified, interactive view of resources, users and their interactions over time [15].
- Prebuilt visualizations for AWS accounts, EC2 instances, users, roles, IP addresses and GuardDuty findings. Each panel includes text explaining how to read it [15].
- Finding groups: a built-in ML model groups related findings, entities and evidence, so one possible incident is investigated together instead of finding by finding [15].
- Generative AI summaries that describe a finding group's chain of events in natural language [13].
- Detective evidence: informational findings drawn from the last 45 days of graph data that highlight unusual activity, such as newly observed geolocations or API calls [15].
- Automated investigations of IAM users and roles for indicators of compromise such as impossible-travel logins, known bad IP addresses and a history of security findings. They can also show which tactics, techniques and procedures (TTPs) a role triggered [13].
- Baseline comparison: recent activity is compared with historical baselines, which are established after two weeks of monitoring [15].
- Cross-service workflows: links from the GuardDuty and Security Hub consoles open a curated Detective page for the selected finding, and partner SOAR products can link to Detective pages inside their playbooks [13].
- Security Lake integration: Detective generates prebuilt Amazon Athena SQL queries scoped to the time and entity under investigation, so analysts can preview or download the underlying CloudTrail and VPC Flow Logs [13].

Architecture and Working Mechanism

- An administrator account enables Detective in a region, which creates a behavior graph. Member accounts join by invitation, or through AWS Organizations via a delegated administrator [13].
- Detective ingests data through its own independent log streams, so it does not affect workload performance or the cost of the source services. It extracts time-based events such as login attempts, API calls and network traffic [13].
- It organizes the data into a graph of findings, entities and relationships. Entity properties come directly from source records or are aggregated across records, and the graph updates continuously [13].
- Machine learning, statistical analysis and graph theory build behavior baselines and group related findings. Analysts then explore the graph through the console visualizations, finding groups and investigations, or through the Detective API [15].

- For raw evidence, the optional Security Lake integration (deployed with a CloudFormation template) lets Detective query CloudTrail and VPC Flow Logs through Athena [13].
- Data stays within the region where Detective is enabled. Disabling Detective deletes the behavior graph and cannot be undone [13].

Data Sources

- AWS CloudTrail logs: API calls and login activity. Core source.
- Amazon VPC Flow Logs: network traffic between resources. Core source [15].
- Amazon GuardDuty findings: starting points for investigations. GuardDuty does not have to be enabled for Detective to work [15].
- AWS Security Hub findings: findings from integrated services such as Inspector, AWS Config and IAM Access Analyzer. Optional; on by default for new accounts [15].
- Amazon EKS audit logs: Kubernetes user and configuration activity. Optional; audit-log coverage is for EKS on EC2 [15].
- Amazon Security Lake: retrieval of raw CloudTrail and VPC Flow Logs. This is a query integration, not a source for the graph [13].

AI/ML and Automation Capabilities

- Machine learning and graph analysis build the behavior graph and the baselines used to spot unusual activity [13].
- A built-in ML model groups related findings into finding groups [13].
- Generative AI writes natural-language summaries of finding groups [13].
- ML-based automated investigations query the graph to decide whether an IAM user or role shows anomalous behavior or indicators of compromise [13].
- Automation hooks include the Detective API, partner SOAR integrations and the prebuilt Athena queries [13].

The AWS documentation reviewed describes AI that supports the analyst (grouping, summaries, IAM investigations). It does not describe a free-form natural-language question interface or automatically generated incident reports. AWS's separate GuardDuty investigation agent, currently in preview [16], covers automated triage of GuardDuty findings.

Strengths

- Nothing to deploy: no agents or log feeds, and no impact on workload performance.
- Long memory: up to a year of aggregated activity, with baselines for comparison.
- Less triage effort: finding groups replace finding-by-finding review.
- Guided and visual: each visualization explains how to interpret it, which helps less experienced analysts.
- Native integration with GuardDuty, Security Hub, Security Lake and AWS Organizations.
- Pay as you go, with a 30-day free trial and no upfront cost [13].

Limitations

- AWS-only and closed: analysts cannot add their own data sources, models or logic beyond what AWS exposes.
- Regional: it must be enabled region by region, and baselines need about two weeks to form [15].
- Cost grows with data volume, and raw log retrieval through Security Lake adds Athena queries and extra deployed resources, each with charges.
- Limited container coverage: EKS audit-log coverage is for EKS on EC2, with no visibility into non-managed Kubernetes.
- Narrow AI features: grouping, summaries and IAM investigations, with no documented free-form question answering or report generation.
- Permanent deletion: disabling Detective deletes the behavior graph, and it cannot be restored [13].

Relevance to Our Project

Detective is the closest AWS-native counterpart to our AI Cloud Security Investigator: it investigates AWS activity from CloudTrail and VPC Flow Logs, correlates findings and summarizes them.

- Ideas to adopt: model accounts, roles, IP addresses and resources as a graph, compare current activity with a baseline, group related findings into one incident, and summarize each group in plain language.
- Closest overlap with our design: Detective's prebuilt Athena SQL queries, scoped to an entity and time window, resemble our natural-language-to-read-only-SQL component, but the analyst cannot ask free-form questions.
- Where our project can differ: free-form natural-language questions answered with supporting evidence, next-step suggestions during an incident, and generated incident reports, none of which the AWS documentation reviewed describes.
- Possible integration: Detective findings and graph data could be one more input to our system, a design option for the team to evaluate.

2.3 Amazon Security Lake

Product Overview

Amazon Security Lake is a fully managed AWS service that automatically centralizes security data from AWS environments, third-party providers, and other supported sources [17]. It transforms data from natively supported AWS services into the Open Cybersecurity Schema Framework (OCSF) format and stores it in Amazon S3 [18], making security information easier to access and analyze across different systems.

Main Purpose

Security Lake helps organizations collect, centralize, and standardize security data from multiple sources. Its main purpose is to provide a unified data environment that supports

security monitoring, threat detection, and incident investigation through integrated analytics tools [17].

Main Features

- Centralized Data Collection: Automatically collects security logs and events from supported AWS services and third-party sources into one centralized data lake [17].
- Data Normalization: Uses the Open Cybersecurity Schema Framework (OCSF) to standardize supported security data, making it easier to analyze across different sources [18].
- Scalable Storage: Stores security data in Amazon S3 using Apache Parquet format for efficient storage and querying [17].
- Multi-Account and Multi-Region Support: Allows organizations to centralize security data across multiple AWS accounts and Regions [17].
- Integration with Other Tools: Supports subscribers and analytics services, such as Amazon Athena, to access and analyze collected data [19].
- Data Management: Provides control over data retention, storage, and lifecycle management [17].

Architecture and Working Mechanism

Amazon Security Lake works through the following main steps:

- Data Collection: Collects security logs and findings from supported AWS services and third-party sources [2].
- Data Transformation: Automatically transforms data from natively supported AWS services into OCSF and Apache Parquet formats [18]. Custom sources must prepare their data according to Security Lake's required format and schema before sending it [20].
- Data Storage: Stores the normalized data in the organization's Amazon S3 buckets using Apache Parquet format.
- Data Access: Makes the collected data available to subscribed services and analytics tools [19].
- Security Analysis: External tools can query and analyze the stored data to identify suspicious activities and investigate security events [19].

Data Sources

Amazon Security Lake supports several security data sources, including [2], [21]:

- AWS CloudTrail: Records account activity and API calls made within AWS.
- Amazon VPC Flow Logs: Provides information about network traffic entering and leaving VPC network interfaces.
- Amazon Route 53 Resolver Query Logs: Records DNS queries made from resources within a VPC.
- Amazon EKS Audit Logs: Records activities performed through the Kubernetes API server.

- AWS Security Hub CSPM Findings: Provides security findings related to cloud security posture and configuration.
- AWS WAF Logs: Records web requests processed by AWS Web Application Firewall.
- Third-Party Sources: Supports security data from compatible external tools and custom sources [20].

AI/ML and Automation Capabilities

Amazon Security Lake primarily focuses on automated security data collection, transformation, normalization, and storage rather than independently performing the complete threat detection and investigation process. However, it supports integration with other AWS services, including Amazon Bedrock and Amazon SageMaker AI, which can generate AI-powered insights from Security Lake data [22]. Therefore, its AI-related capabilities depend on the services and tools integrated with the data lake.

Strengths

- Centralized Visibility: Combines security data from multiple sources into one environment.
- Standardized Data: Uses OCSF to make security information more consistent and easier to analyze.
- Scalability: Uses Amazon S3 to support large volumes of security data.
- Multi-Account Support: Helps organizations manage security data across multiple AWS accounts and regions.
- Integration: Allows external analytics and security tools to access the collected data.
- Flexible Analysis: Enables organizations to use different tools to query and investigate stored security information.

Limitations

- Dependence on External Investigation Tools: Does not independently correlate events or reconstruct complete attack timelines [17].
- Dependence on Integrated AI/ML Services: Requires external tools for advanced machine-learning-based threat detection and analysis [22].
- Configuration Requirements: Data sources and subscriber integrations must be configured correctly.
- AWS-Centered Integrations: Its main functionality is designed around AWS infrastructure and supported integrations.
- Storage Costs: Storing large volumes of security logs over long periods can increase storage and analysis costs.
- Data Quality Dependency: The quality of security analysis depends on the availability and completeness of the collected data.

Relevance to Our Project

Amazon Security Lake is relevant to our AI Cloud Security Investigator because it provides a centralized environment for collecting and standardizing cloud security data. It can support

the data collection stage of our project by bringing information from sources such as CloudTrail and VPC Flow Logs into a common format. However, our project aims to go further by correlating events across services, reconstructing attack timelines, mapping activities to MITRE ATT&CK, and generating investigation reports using AI. Therefore, Security Lake can serve as a useful data foundation, while our proposed system focuses on automated investigation and analysis.

2.4 Comparison of Market Products

The tables below compare the three products reviewed in this section. The AI features listed are those described in the public documentation cited in each subsection.

Aspect	AWS Security Incident Response	Amazon Detective	Amazon Security Lake
Main role	Managed incident response: triage, AI investigation, containment and 24/7 expert support	Investigating suspicious activity and finding root causes	Collecting and standardizing security data
Main data sources	GuardDuty and third-party findings via Security Hub; CloudTrail, IAM, EC2 APIs and Cost Explorer for the AI investigation	CloudTrail, VPC Flow Logs, GuardDuty, Security Hub, EKS audit logs	CloudTrail, VPC Flow Logs, Route 53 Resolver logs, EKS audit logs, Security Hub CSPM, WAF logs, third-party sources
Event correlation and timeline	Yes: the AI agent correlates evidence and builds a timeline	Yes: behavior graph and ML-based finding groups	No: does not correlate events or rebuild attack timelines by itself
AI/ML used	Agentic AI investigation from a natural-language incident description	ML, statistical analysis and graph theory; generative AI summaries	None itself; integrates with Amazon Bedrock and SageMaker AI
Evidence behind each statement	Not described	Prebuilt Athena SQL queries scoped to an entity and time window	Not applicable
MITRE ATT&CK mapping	Not described	Shows the TTPs a role triggered	No
Incident report	AI summary with timeline and recommendations	Finding-group summaries; no automatic incident reports documented	No

Aspect	AWS Security Incident Response	Amazon Detective	Amazon Security Lake
Containment	Yes: run on the customer's behalf with permission	No	No
Access and scope	AI agent needs AWS Organizations and works only on AWS-supported cases	AWS-only, closed and regional	AWS-centered integrations
Pricing model	Per finding ingested; first 10,000 per month free	Per GB ingested; 30-day free trial	Storage and analysis costs grow with data volume
Main limitation	Closed model and prompts; no published accuracy figures	Narrow AI features; cost grows with data volume	Needs external tools for investigation and ML

Gaps Identified in Existing Products

- No guarded free-form questions over the logs. The documentation reviewed describes one AI summary per case in AWS Security Incident Response, and grouping, summaries and IAM investigations in Amazon Detective. Detective's prebuilt Athena queries are scoped to an entity and time window, so analysts cannot ask their own questions. Microsoft Security Copilot [23], Google SecOps [24] and the Splunk AI Assistant for SPL [25] do translate plain-English questions into queries, but each uses its own query language. None of the documentation reviewed describes a read-only guard on generated queries.
- No evidence trail behind each statement. The documentation reviewed does not describe report statements that link to the specific log events supporting them, which an investigator needs to verify an AI-written finding.
- No published reliability measures. AWS Security Incident Response publishes no accuracy figures, and AWS states that users must verify AI findings themselves [11]. None of the documentation reviewed reports how often the AI produces statements that the logs do not support.
- Limited correlation across sources. Amazon Security Lake does not correlate events or reconstruct attack timelines on its own [17], and most of the research papers in Section 3 analyze single events or a single service (for example, S3 only). Few of the works reviewed combine multi-service correlation, ATT&CK mapping and report generation.

3. Research Survey

3.1 Retrieval-Augmented Large Language Model for AWS Cloud Threat Detection and Modelling: CloudTrail MITRE ATT&CK Mapping

Aspect	Details
Paper information	G. Adediran, K. Awuson-David and Y. Ahmed, Birmingham City University and De Montfort University, UK. Computers, Materials & Continua (Tech Science Press), published online 28 February 2026. DOI: 10.32604/cmc.2026.077606 [3]
Problem statement	CloudTrail produces huge event volumes, and SIEM rules and signature-based detection cause alert fatigue, high false-positive rates and missed novel attacks. Querying CloudTrail Lake needs strong SQL skills. ML detectors need large labeled datasets and become obsolete as threats change, and a plain LLM relies on out-of-date training data and can hallucinate detections.
Objectives	Improve CloudTrail threat detection by grounding an LLM in external threat intelligence, so that each event is mapped to a MITRE ATT&CK technique, classified with STRIDE and given a severity, together with a natural-language explanation and remediation advice that analysts can verify.
Methodology	Two-step RAG pipeline on Google Vertex AI. Step 1: Gemini 2.5 Pro reads the raw CloudTrail JSON event and writes a natural-language search query summarizing its security-relevant attributes. Step 2: the query retrieves the top-k most similar chunks from a vector database, and Gemini uses them in a RAG-grounded prompt to produce the ATT&CK mapping, STRIDE classification, severity and remediation advice. The knowledge base holds the MITRE ATT&CK Enterprise Cloud matrix, the AWS Threat Technique Catalogue, cloud threat reports and security blogs. The backend is deployed on Google Cloud Run.
Dataset	1,724 CloudTrail events generated with Stratus Red Team (20 of its 44 AWS attack techniques, run in a sandbox account). 200 events were chosen by stratified sampling (seed 42): 122 malicious and 78 benign, covering 9 ATT&CK techniques, 8 tactics and 9 AWS services. Ground truth was labeled by a cybersecurity expert using a four-step guideline (context analysis, ATT&CK mapping, malicious or benign, STRIDE and severity). Each event is analyzed on its own, with no sequences.
AI/ML techniques	Gemini 2.5 Pro (up to 1,048,576 input tokens) as the LLM. Retrieval uses text-multilingual-embedding-002 embeddings (1,024-token chunks with 256-token overlap) stored in the Vertex AI RAG vector database. Prompting combines few-shot, hidden chain-of-thought, role, structured-output and evidence-grounded prompts, with prompt chaining between the two steps. Threat knowledge is updated by refreshing the knowledge base instead of retraining the model.

Aspect	Details
Evaluation	Accuracy, precision, recall and F1 (scikit-learn) of the ATT&CK technique mapping against the expert labels, comparing the RAG system with baseline Gemini 2.5 Pro without retrieval on the same 200 events. Also a confusion matrix, accuracy by tactic and by severity, error categorization, STRIDE confidence levels, two case studies, and cost and latency per event.
Results	RAG vs baseline: accuracy 78% vs 46%, precision 85% vs 69%, recall 78% vs 46%, F1 79% vs 45% (76.4% relative F1 gain). Accuracy by tactic: 100% for Credential Access, Persistence, Privilege Escalation, Collection and Initial Access; 87% for Discovery; 66.7% for Defense Evasion; 57.1% for Exfiltration. Critical events 100% and High events 97.2% correct; benign events 87.2% correct (12.8% false positives). Cost and latency: $0.00376 and 4.1 s per event vs $0.00163 and 1.5 s for the baseline (2.3 times the cost, 2.7 times the latency). Errors fall into retrieval-generation gaps, knowledge base gaps and ambiguous ground truth.
Limitations	Single events only, so multi-step attacks and attack timelines are not covered; Tampering and Elevation of Privilege were mostly low confidence for this reason. Attacks are simulated by one tool in one sandbox, and malicious labels rest on the “stratus-redteam” user-agent marker. Only a plain LLM is compared; no SIEM, GuardDuty or ML baseline was run, so the claim of cost comparable to commercial SIEMs is not measured. The reported numbers are inconsistent: F1 is 79% in the abstract but 80% in the contributions; retrieval-generation errors are 60% in the contributions and conclusion but 26% in Section 6.4; Tables 3, 5 and 6 add up to 167 malicious and 245 total events, not 122 and 200; the abstract states 90% inter-annotator agreement, yet one annotator is described. No direct measure of hallucination.
Project relevance	High: it works on the same data type as our project (AWS CloudTrail events) and shows that grounding an LLM in external threat knowledge raises accuracy a lot over a plain LLM. Its ATT&CK and STRIDE mapping can be reused in our reports, and its error categories (retrieval errors, missing knowledge, ambiguous events) tell us what to test. Our project differs: analysts ask follow-up questions in plain English that are translated to SQL over the logs, we look at sequences of events instead of single ones, and we measure how often the LLM makes unsupported claims, which this paper does not measure directly.

3.2 Interpretable AI for Smart City Cloud Security: An MCP Framework for Real-Time IoT Threat Detection

Aspect	Details
Paper information	A. Aldhamari and S. Zhou, University of Portsmouth, UK. Urban Science (MDPI), vol. 10, art. 378, published 2 July 2026. DOI: 10.3390/urbansci10070378 [6]

Aspect	Details
Problem statement	ML threat detectors are “black boxes”. Explanation methods such as SHAP and LIME give abstract feature weights, are slow, and are not understandable by non-technical decision makers.
Objectives	Detect threats in cloud logs in real time and explain each alert in plain language, adapted to the reader (security analyst, administrator or public-safety official).
Methodology	CRISP-DM process. Events are normalized to a CloudTrail-like JSON schema, features are extracted, a classifier flags suspicious events, and for high-confidence alerts a Model Context Protocol (MCP) layer builds a structured JSON context document that an LLM turns into a narrative. Integrated with Elasticsearch (SIEM).
Dataset	flaws.cloud: 1.9 million real AWS CloudTrail events, including a 1.3 million-event cryptomining campaign. Labels are heuristic (rules such as failed logins, API bursts, night-time activity), with a 4:1 benign-to-suspicious ratio.
AI/ML techniques	Random Forest (50 trees, depth 6, balanced class weights, 5-fold cross-validation) on engineered temporal, identity, geographic and behavioral features. GPT-4 at temperature 0.3 with role-specific prompts generates the explanations.
Evaluation	Stratified 70/30 random split (570,000 test events); accuracy, precision, recall, F1, ROC-AUC and calibration (ECE); comparison with Logistic Regression and Gradient Boosting; feature-group ablation; manual review of 200 generated narratives.
Results	Accuracy 84.2%, precision 48.2%, recall 73.3%, F1 58.1%, AUC 0.884. 96.8% of the cryptomining events detected, first alert after 4.2 s. Detection under 13 ms per event; explanation about 2.3 s. Removing temporal features cost 9.3 points of accuracy.
Limitations	Labels come from rules, not confirmed attacks, and some features overlap with those rules. Precision of 48.2% means more than half of the alerts are false positives; with a 4:1 class ratio, labeling everything benign already scores 80%. Random rather than time-based split. No user study, and manual review found narratives overstating impact in 12% of samples. GPT-4 costs an estimated $3,000–5,000$ per month. Adversarial robustness not tested. The paper states 20 features but its ablation study reports 35.
Project relevance	Very high: it uses the same flaws.cloud CloudTrail dataset we use, and the same idea of an LLM explaining security events from structured evidence that cites event fields. Our project extends it: the analyst asks follow-up questions in plain English (translated to SQL over the logs) instead of receiving one fixed narrative, and we measure how often the LLM makes unsupported claims, which this paper does not.

3.3 Interpretable Attack-Chain Stage Detection from AWS CloudTrail Event Sequences via Linear Models and HMM Smoothing

Aspect	Details
Paper information	Jingwen Bai et al., INFOTRON, Vol. 6, No. 1, 2026, pp. 28–43 [4].
Problem statement	Analyzing CloudTrail events separately makes it difficult to understand how an attack develops through different stages.
Objectives	Classify CloudTrail events into attack stages and reconstruct a more consistent attack sequence.
Methodology	TF-IDF extracts event features. LinearSVC and Logistic Regression classify events into MITRE ATT&CK tactics. HMM with the Viterbi algorithm improves sequence consistency.
Dataset	310 AWS CloudTrail events generated using Stratus Red Team, covering 35 attack techniques and 10 MITRE ATT&CK tactics.
AI/ML techniques	TF-IDF, LinearSVC, Logistic Regression, and Hidden Markov Models (HMM).
Evaluation	Classification accuracy, macro-F1, and attack-chain boundary detection.
Results	LinearSVC achieved 96.1% accuracy and 0.866 macro-F1. HMM improved attack-chain boundary F1 from 0.800 to 0.892.
Limitations	Small simulated dataset; focuses on CloudTrail sequences rather than correlating multiple cloud services.
Project relevance	Both projects use cloud events to understand attack progression. Our project extends this by correlating multiple cloud sources, building attack timelines, mapping MITRE ATT&CK techniques, and generating investigation reports.

3.4 An Explainable Machine Learning Framework for AWS S3 CloudTrail-Like Log Analysis in Digital Forensics

Aspect	Details
Paper information	Title: An Explainable Machine Learning Framework for AWS S3 CloudTrail-Like Log Analysis in Digital Forensics. Authors: A. Aljuhani, A. A. Alhashmi and A. A. Darem. Journal: International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 17, No. 10, pp. 122–130, 2026 [1]
Problem statement	Cloud investigators cannot access physical media and depend on audit logs such as CloudTrail. The volume, speed and variety of these logs make manual analysis slow. Existing ML detectors focus on accuracy and say little about why an event was flagged, which forensic work requires.

Aspect	Details
Objectives	Build an explainable hybrid ML pipeline that flags suspicious activity in S3 CloudTrail-style logs (unauthorized access, data exfiltration, destructive deletion), explains each alert, and follows forensic principles (NIST SP 800-86, ISO/IEC 27037).
Methodology	Raw events are parsed into canonical operations (LIST, PUT, GET, DELETE, RM). 63 features are engineered, including sensitive-keyword flags, an exfiltration-bucket flag, time encodings, and rolling counts over 1, 5, 10, 15 and 30 minutes. XGBoost (calibrated) and Isolation Forest scores are blended (0.8 / 0.2) and raised by a bounded rule-based boost of up to 0.12. SHAP explains the XGBoost predictions. A chronological split (FIT/CAL/TUNE/TEST) with 30-minute embargoes limits data leakage, and there are two decision thresholds: monitoring (0.6624) and forensic (0.7003).
Dataset	169,986 synthetic CloudTrail-like JSONL events from an AWS-like lab, with normal and attack behavior. Labels were generated by heuristic rules, with 15.95% suspicious. The main evaluation used 35,963 temporally held-out events.
AI/ML techniques	XGBoost (supervised classifier), Isolation Forest (unsupervised anomaly detection), sigmoid probability calibration, rule-based reasoning, SHAP (explainable AI), and temporal behavior analysis. Random Forest and Logistic Regression served as baselines.
Evaluation	Metrics: PR-AUC, ROC-AUC, precision, recall and F1. Experiments: ablation of feature groups, three chronological cross-validation folds, a 50,000-event noise perturbation test, SHAP stability tests, baseline comparisons, and a comparison with two Elasticsearch rule queries.
Results	Held-out PR-AUC 0.9324 and ROC-AUC 0.9827. Monitoring mode: precision 0.9677, recall 0.7930, F1 0.8717. Forensic mode: precision 0.9822, recall 0.7844, F1 0.8722. Removing sensitive-object features dropped monitoring F1 to 0.6108. SHAP top-ranked features were stable in 97.33% of perturbation pairs. XGBoost-only scored comparable F1 (0.8723), so the framework's value is mainly its integrated, explainable workflow rather than higher accuracy.
Limitations	Synthetic data only, and the labels overlap some of the engineered features, so scores may reflect labeling conventions. It covers only S3 object activity, with no IAM, VPC Flow Logs or CloudWatch data. SHAP explains only the XGBoost component, not the full hybrid score. There was no user study with forensic investigators. The anomaly component showed no measured predictive gain.

4. Proposed Innovative Solution

4.1 Novel Capabilities Beyond Existing Literature

After reviewing existing cloud security products and the research discussed in the previous sections, we identified several areas where the investigation process could be improved. Our proposed system aims to bring these capabilities together in one workflow, with a focus on making cloud security investigations easier to perform, understand, and verify.

- **Conversational Natural-Language Querying:** Security analysts often need to write queries or navigate predefined dashboards to investigate suspicious activity. For example, Amazon Detective mainly provides predefined visualizations and investigation features [13]. Our proposed system aims to make this process more flexible by allowing analysts to ask questions about cloud logs in natural language. These questions can then be translated into read-only SQL queries, allowing analysts to investigate events without having to write SQL manually.
- **Transparency of Generated Queries:** When an AI system generates an answer, analysts should be able to understand how that answer was produced. For this reason, our system is designed to display the SQL query used to retrieve the relevant log data. This gives analysts an opportunity to review the query and check whether the returned information supports the generated response.
- **Linking Findings to Supporting Evidence:** Investigation reports are more useful when their conclusions can be traced back to the original evidence. The work by Adediran et al. [3] demonstrates the use of retrieved threat knowledge to support CloudTrail analysis. Building on this idea, our proposed system aims to link investigation findings to the relevant CloudTrail records, making it easier for analysts to verify the information instead of relying only on AI-generated explanations.
- **Attack Timeline Reconstruction and MITRE ATT&CK Mapping:** Individual log events may not provide enough information to understand an attack. Looking at their order and relationships can help reveal how an incident developed. Previous research has explored attack-sequence analysis using Hidden Markov Models [4], while other work has focused on classifying individual CloudTrail events [1]. Our system aims to combine event correlation and timeline reconstruction with MITRE ATT&CK mapping [5] to provide a clearer view of suspicious activities.
- **Risk-Based Event Prioritization:** Not all security events require the same level of attention. A system that treats every alert equally may make it harder for analysts to focus on the most important incidents. Our proposed approach considers the potential impact of different classification errors when prioritizing events for further investigation. This is intended to help analysts focus on events that may represent a greater security risk.
- **Flexible CloudTrail Log Ingestion:** The system is designed to work with CloudTrail logs from individual AWS accounts as well as previously exported log files. This provides flexibility for testing and investigation, since users do not have to rely on a single live AWS environment to analyze security events.

- **Evaluation of Unsupported AI-Generated Findings:** AI-generated explanations should not be treated as reliable simply because they sound convincing. In addition to evaluating detection and analysis performance, our project aims to examine whether the generated findings are supported by the available log evidence. This can help identify cases where the language model produces information that cannot be verified from the source records.

Together, these capabilities aim to provide a more connected investigation process, combining automated event analysis, contextual explanations, and evidence that analysts can review.

4.2 Anticipated Advantages and Operational Constraints

Anticipated Advantages

Our proposed system is expected to provide several benefits during cloud security investigations:

- **Faster Investigations:** Automating log parsing and event correlation can reduce the time analysts spend manually reviewing large amounts of security data.
- **Easier Verification of Results:** By displaying the SQL queries and linking findings to their original log records, the system allows analysts to check the information behind each AI-generated result.
- **More Flexible Investigation:** Analysts can ask questions in natural language instead of relying only on predefined queries or dashboard views. This can make it easier to investigate specific activities or explore suspicious events further.
- **Measurable Performance:** Testing the system using benchmark security datasets will help us evaluate its ability to detect suspicious activities, reconstruct attack timelines, and produce explanations supported by the available evidence.

4.3 Overview of Analytical Methods

Our proposed system uses supervised classification as the main approach for identifying suspicious security events. Since the system works with labeled security data, it can learn from previously identified malicious and benign events and use this knowledge when analyzing new events.

We selected classification because the events we want to identify belong to specific categories, such as normal activity and different types of attacks. This also allows us to evaluate the model using measurable metrics such as precision, recall, and F1-score.

Our decision is also supported by the research papers reviewed in Section 3:

- Aldhamari and Zhou [6] used a Random Forest classifier to identify suspicious CloudTrail events before generating explanations using an LLM.
- Bai et al. [4] used LinearSVC and Logistic Regression to classify events into MITRE ATT&CK tactics, followed by an HMM to improve the consistency of attack sequences.

- Aljuhani et al. [1] used XGBoost to identify suspicious S3 activities and applied SHAP to explain the model's predictions.
- Adediran et al. [3] used a RAG-based approach to classify CloudTrail events and map them to MITRE ATT&CK techniques.

Based on these studies, we chose to use classification as the foundation of our detection process and XGBoost as our primary classifier. We also use RAG as an additional component to provide threat-related context and generate understandable explanations of the identified activities.

4.4 Classification with the Bayesian Decision Extension.

Overview:

Classification is a machine learning technique that sorts new items into categories after learning from examples that are already labeled. In our project, the items are security events, and the categories are normal activity and different types of attacks.

The Bayesian extension adds one more idea: not every mistake costs the same. First, a model estimates how likely an event is to be an attack. Then Bayesian decision theory combines that probability with the cost of each possible mistake and picks the action with the lowest expected cost, called the minimum expected risk. In simple terms, missing a real attack is far worse than reviewing a harmless event, so the system should investigate even when an event is only fairly suspicious. The risk of each action is calculated as:

$R(\text{action} \mid \text{event}) = \Sigma\ \text{cost}(\text{action}, \text{class}) \times P(\text{class} \mid \text{event})$

The system chooses the action with the smallest risk.

Implementation Role

XGBoost provides the probability that each event is an attack. We define two actions, Ignore and Investigate, with this starting cost matrix:

Action	True class	Benign Attack
Ignore	0	10
Investigate	1	0

With these costs, the system investigates any event whose attack probability is above 1/11, about 0.09, instead of using a plain 0.5 cutoff. Events selected for investigation are passed to the RAG component . The cost values are starting points that we will tune and test.

Why We Chose It

- **Mistakes are not equal**: a missed attack can be serious, while a false alarm only costs analyst time. A fixed 0.5 cutoff ignores this difference.
- **Clear and explainable decisions**: the threshold comes directly from stated costs, so it is easy to adjust and to explain to analysts.
- **It delivers our risk-based prioritization capability** and limits expensive RAG processing to the events that matter.
- **It fits our data**: attack types are discrete categories, labeled examples are available, and results can be measured with precision, recall and F1-score. The reviewed papers [1], [3], [4], [6] also treat detection as a classification task.
- **It is the best fit among the lecture techniques**: regression predicts numbers, PAC learning gives theoretical bounds rather than a working detector, parametric learning assumes a fixed data distribution, and clustering is unsupervised.

Limitations

The decision rule is only as good as the probabilities it receives. Aljuhani et al. [1] calibrated their XGBoost scores, and we will check ours on validation data. The cost values are judgments, so we will test how results change when they change. The model also depends on the quality of its labeled data, may miss attack types it has not seen, and faces class imbalance, so we will report macro-F1 instead of accuracy alone.

4.5 XGBoost with SHAP (Aljuhani et al.)

Overview

XGBoost builds many small decision trees one after another, where each new tree focuses on the mistakes of the previous ones. The final answer combines all the trees [26]. It works like a panel of reviewers in which each member concentrates on the cases the earlier members got wrong. SHAP is a companion method that shows which features pushed a particular prediction toward "attack" or "normal" [27].

Implementation Role

XGBoost is the model that performs the classification. For each event, it receives features such as the API call, the event type, and how many events the same user or source produced in the last few minutes. It outputs the probability of each class, which feeds the Bayesian decision rule in Section 4.4. For events flagged as suspicious, SHAP shows the analyst which features mattered most. We also train a simpler Random Forest on the same data as a comparison baseline, as Aljuhani et al. [1] and Aldhamari and Zhou [6] did, to show whether XGBoost adds value.

Why We Chose It

- **It suits our data**: security logs become structured tables of features, which is where tree-based models perform best.
- **It captures combinations of signals**, such as an unusual API call at an unusual time, that simpler models can miss.
- **It is proven on similar data:** Aljuhani et al. [1] reported a PR-AUC of 0.9324 on time-held-out AWS log events, and their SHAP top features stayed stable in 97.33% of perturbation tests.
- **It produces probabilities**, which the Bayesian decision step needs.
- It is fast, so it can screen every event cheaply before the slower RAG step.
- With SHAP, its decisions can be explained, which investigations require.

Limitations

Aljuhani et al. [1] used synthetic S3-only data with rule-generated labels, so results may differ on other AWS services or real environments. XGBoost also needs good labels, careful feature design and parameter tuning.

4.6 Retrieval-Augmented Generation (Adediran et al.)

Overview

Retrieval-Augmented Generation (RAG) lets a large language model (LLM) look up information in an external knowledge base before it answers [28]. It works like an open-book exam: instead of relying only on memory, the model checks relevant reference material first, which makes its answers more accurate and easier to verify.

Implementation Role

RAG handles the events that the Bayesian rule selected for investigation. It retrieves matching descriptions from the MITRE ATT&CK knowledge base [5], and the LLM uses them to map the activity to an attack technique, explain it in plain language, suggest containment steps, and point to the log records that support its answer.

The same LLM also lets analysts ask questions in everyday language. It translates each question into a read-only SQL query and shows that query to the analyst. Three safeguards keep queries safe: a parser that accepts only single SELECT statements on approved tables, a database account with read-only permission, and a limit on rows returned.

Why We Chose It

- **It improves accuracy:** Adediran et al. [3] raised ATT&CK mapping accuracy from 46% to 78% by adding retrieval to the same model.
- **It stays current:** the knowledge base can be updated when new threats appear, without retraining the model.
- **Answers are grounded and checkable**, which supports our goals of linking findings to evidence and measuring unsupported claims.
- **It gives analysts readable explanations**, where XGBoost and SHAP give only numbers and feature rankings.
- **It works on our data type**: the reviewed study used AWS CloudTrail events [3].

Limitations

Adediran et al. [3] report about 4.1 seconds and $0.00376 per event, so running RAG on every log entry would be slow and costly. We apply it only to events selected by the Bayesian rule. RAG can still produce wrong explanations, so every finding is linked to its log records, and we will measure how often claims are unsupported. The reviewed study also analyzed single events, not sequences.

How the Techniques Work Together

XGBoost estimates how likely each event is to be an attack, the Bayesian rule decides whether it is worth investigating, and RAG explains the selected events and links them to evidence. Each technique does a different job, which keeps the system fast on large volumes of logs while still giving analysts explanations they can check.

References

1. A. Aljuhani, A. A. Alhashmi and A. A. Darem, “An Explainable Machine Learning Framework for AWS S3 CloudTrail-Like Log Analysis in Digital Forensics,” *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 17, no. 10, pp. 122–130, 2026.
2. Amazon Web Services, “Collecting Data from AWS Services,” *Amazon Security Lake User Guide*. https://docs.aws.amazon.com/security-lake/latest/userguide/internal-sources.html
3. G. Adediran, K. Awuson-David and Y. Ahmed, “Retrieval-Augmented Large Language Model for AWS Cloud Threat Detection and Modelling: CloudTrail MITRE ATT&CK Mapping,” *Computers, Materials & Continua*, Feb. 2026, doi: 10.32604/cmc.2026.077606. https://doi.org/10.32604/cmc.2026.077606
4. J. Bai et al., “Interpretable Attack-Chain Stage Detection from AWS CloudTrail Event Sequences via Linear Models and HMM Smoothing,” *INFOTRON*, vol. 6, no. 1, pp. 28–43, 2026.
5. MITRE, “Cloud Matrix,” *MITRE ATT&CK Enterprise Matrices*. https://attack.mitre.org/matrices/enterprise/cloud/
6. A. Aldhamari and S. Zhou, “Interpretable AI for Smart City Cloud Security: An MCP Framework for Real-Time IoT Threat Detection,” *Urban Science*, vol. 10, art. 378, Jul. 2026, doi: 10.3390/urbansci10070378. https://doi.org/10.3390/urbansci10070378
7. Amazon Web Services, “AWS announces AWS Security Incident Response for general availability,” Dec. 2024. https://aws.amazon.com/about-aws/whats-new/2024/12/aws-security-incident-response/
8. Amazon Web Services, “AWS Security Incident Response now provides agentic AI-powered investigation,” Nov. 2025. https://aws.amazon.com/about-aws/whats-new/2025/11/aws-security-incident-response-agentic-ai-powered-investigation
9. Amazon Web Services, “AWS Security Incident Response pricing.” https://aws.amazon.com/security-incident-response/pricing/ (accessed Nov. 3, 2026).
10. Amazon Web Services, “AWS Security Incident Response features.” https://aws.amazon.com/security-incident-response/features/ (accessed Nov. 3, 2026).
11. Amazon Web Services, “AI Investigative Agent,” *AWS Security Incident Response User Guide*. https://docs.aws.amazon.com/security-ir/latest/userguide/ai-investigative-agent.html (accessed Nov. 3, 2026).
12. Amazon Web Services, “AWS Security Incident Response FAQs.” https://aws.amazon.com/security-incident-response/faqs/ (accessed Nov. 3, 2026).
13. Amazon Web Services, “Amazon Detective FAQs.” https://aws.amazon.com/detective/faqs
14. Tutorials Dojo, “Amazon Detective.” https://tutorialsdojo.com/amazon-detective/
15. Amazon Web Services, “Amazon Detective concepts and terminology,” *Amazon Detective User Guide*. https://docs.aws.amazon.com/detective/latest/userguide/detective-terms-concepts.html
16. Amazon Web Services, “Introducing the Amazon GuardDuty investigation agent: On-demand AI-powered threat assessment,” *AWS Security Blog*, Jul. 20, 2026. https://aws.amazon.com/blogs/security/introducing-the-amazon-guardduty-investigation-agent-on-demand-ai-powered-threat-assessment/ (accessed Nov. 18, 2026).
17. Amazon Web Services, “What is Amazon Security Lake?,” *Amazon Security Lake User Guide*. https://docs.aws.amazon.com/security-lake/latest/userguide/what-is-security-lake.html

18. Amazon Web Services, “Open Cybersecurity Schema Framework (OCSF),” *Amazon Security Lake User Guide*. https://docs.aws.amazon.com/security-lake/latest/userguide/open-cybersecurity-schema-framework.html
19. Amazon Web Services, “Subscriber Management in Security Lake,” *Amazon Security Lake User Guide*. https://docs.aws.amazon.com/security-lake/latest/userguide/subscriber-management.html
20. Amazon Web Services, “Collecting Data from Custom Sources,” *Amazon Security Lake User Guide*. https://docs.aws.amazon.com/security-lake/latest/userguide/custom-sources.html
21. Amazon Web Services, “Source Management in Security Lake,” *Amazon Security Lake User Guide*. https://docs.aws.amazon.com/security-lake/latest/userguide/source-management.html
22. Amazon Web Services, “AWS Service Integrations with Security Lake,” *Amazon Security Lake User Guide*. https://docs.aws.amazon.com/security-lake/latest/userguide/aws-integrations.html
23. Microsoft, “Microsoft Security Copilot in advanced hunting,” *Microsoft Learn*. https://learn.microsoft.com/defender-xdr/advanced-hunting-security-copilot (accessed Nov. 18, 2026).
24. Google Cloud, “Generate search queries with Gemini,” *Google Security Operations Documentation*. https://docs.cloud.google.com/chronicle/docs/investigation/generate-udm-search-queries-gemini (accessed Nov. 18, 2026).
25. Splunk, “Use Splunk AI Assistant for SPL,” *Splunk Documentation*. https://help.splunk.com/en/splunk-cloud-platform/search/splunk-ai-assistant/1.1.0/use-splunk-ai-assistant-for-spl (accessed Nov. 18, 2026).
26. T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, San Francisco, CA, USA, 2016, pp. 785–794, doi: 10.1145/2939672.2939785.
27. S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*, Long Beach, CA, USA, 2017, pp. 4765–4774.
28. P. Lewis et al., “Retrieval-augmented generation for knowledge-intensive NLP tasks,” in *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020, pp. 9459–9474.