MACHINE LEARNING BASED ANOMALY DETECTION FOR ENHANCED CLOUD SECURITY

AnilKumar J Kadam
Department of Computer Engineering, All India Shri Shivaji Memorial Society's College of Engineering (AISSMS COE), affiliated with Savitribai Phule Pune University, Pune, India

Mrunal Rajendra Sonekar
Department of Engineering, ME Artificial Intelligence and Data Science AISSMS College of Engineering, affiliated with Savitribai Phule Pune University Pune, India,

Abstract: *As cloud computing continues to grow as the core platform for delivering scalable and flexible online services, the need for strong security measures has become more important than ever. Traditional intrusion detection and prevention systems, which rely heavily on fixed attack signatures, often fail to recognize new or advanced threats and frequently raise unnecessary alerts. To overcome these challenges, this study introduces a Machine Learning-Based Anomaly Detection Framework designed to enhance cloud security through adaptive and intelligent analysis. The framework combines several supervised and ensemble learning methods to identify unusual activity within cloud environments and accurately categorize different attack types. It also incorporates a contextual inference process that links system events and traces how faults may spread, helping in early detection of potential issues. Experiments conducted on the UNSW-NB15 and AWS CloudTrail datasets show that the framework delivers strong performance, achieving 97.68% detection accuracy, high precision, and a significant drop in false positives. These outcomes highlight the potential of machine-learning-driven anomaly detection to shift cloud security systems from simple monitoring tools to proactive, real-time defense mechanisms.*

Keywords: **Cloud security, anomaly detection, machine learning, supervised learning, ensemble models, intrusion detection, attack classification, contextual inference, fault propagation, UNSW-NB15, AWS CloudTrail, detection accuracy, false positives reduction, real-time defense**

1. INTRODUCTION

Cloud computing has grown into a fundamental backbone for modern digital services, supporting everything from online applications to large-scale enterprise operations. Its ability to offer on-demand resources, scalability, and cost efficiency has encouraged rapid adoption across industries. However, this widespread use has also increased exposure to security risks. Cloud systems often face diverse and constantly evolving cyber threats, making it difficult for traditional security tools to keep pace. Most conventional intrusion detection methods rely on predefined attack signatures, which limits their ability to identify unfamiliar or sophisticated attacks. They also tend to generate excessive false alerts, creating additional challenges for security teams.

In this context, the need for advanced, adaptable, and data-driven security solutions has become essential. Anomaly detection using machine learning provides a promising direction, as it can learn patterns from real cloud environments, spot unusual activities, and support early threat identification. By analyzing system behavior and correlating different events, such an approach can offer deeper insights into potential attack paths and help uncover the root causes of security issues. With cloud infrastructures becoming more dynamic and complex, integrating intelligent detection mechanisms is crucial for strengthening defense

strategies and ensuring the reliability of cloud services. With cloud infrastructures becoming more dynamic and complex, integrating intelligent detection mechanisms is crucial for strengthening defense strategies and ensuring the reliability of cloud services. As organizations increasingly depend on cloud platforms for critical operations, even a short disruption or unnoticed breach can lead to significant financial loss, data exposure, and damage to user trust. This makes timely detection not just a security requirement but a fundamental part of maintaining uninterrupted service availability.

Moreover, cloud environments often involve distributed resources, multiple access points, and diverse user activities, which can make traditional security methods insufficient. An effective security framework must be capable of understanding the context behind system events, adapting to changing workloads, and identifying abnormal behavior before it escalates. By using advanced analytical techniques to examine traffic patterns, user actions, and system logs, security teams can gain a clearer picture of potential threats and respond more effectively. As cloud services continue to expand and evolve, adopting such proactive and behavior-focused approaches becomes vital for building a safer and more resilient digital ecosystem.

2. RELATED WORK

Research on cloud security has progressed steadily with the growing need to detect threats in large and dynamic cloud environments. Early studies mainly examined rule-based and signature-driven techniques, which struggled to recognize new attack behaviors. To address these limitations, Al-Mazrawe and Al-Musawi [1] provided a broad review of anomaly detection methods in cloud networks, highlighting the shift toward learning-based solutions for more adaptive threat identification. This transition is further reflected in the work of Thapa and Arjunan [2], who surveyed modern machine learning techniques—including neural models, SVMs, clustering, and ensemble strategies—showing their effectiveness in capturing subtle deviations within cloud traffic. One of the foundational contributions in this direction was by Gander et al. [3], who proposed an early Monitoring-as-a-Service model using event correlation and learning-based anomaly detection for public and hybrid cloud setups. As cloud workloads became more diverse, later studies such as Al-Jarrah et al. [4] explored the use of clustering methods, SVM classifiers, and autoencoders to track abnormal behaviors in cloud systems, reinforcing the role of behavior-focused detection rather than rigid rule-based mechanisms. With multi-cloud environments becoming common, Deevi [5] introduced a deep-learning-oriented framework using LSTM and Autoencoder models capable of detecting rare and previously unseen threats in large-scale distributed traffic. In a similar direction, Nwachukwu [6] examined supervised, unsupervised, and reinforcement-based algorithms, emphasizing their ability to reduce detection delays and support early threat mitigation. Broader surveys by Shaker et al. [7] and Babu & Sreelatha [9] further documented the advantages of machine learning–based intrusion detection systems, addressing persistent challenges such as scalability, false alarm reduction, and deployment readiness for real-time cloud scenarios. Recent work has also explored specialized cloud settings and application-specific datasets. Joshi and Shah [8] studied detection mechanisms that combine machine learning with Data Security Posture Management (DSPM), presenting a hybrid strategy for identifying data misuse and cloud configuration risks. Jasani and Padhya [11] focused specifically on AWS CloudTrail logs, demonstrating the strength of supervised learning models in recognizing suspicious activity patterns within commercial cloud platforms. Similarly, Gunupudi and Tamanna [12] analysed machine learning-based intrusion detection for educational cloud environments, reporting promising results from Random Forest and hybrid deep-learning architectures. Mahendar and Shivakanth [10] expanded on these findings by comparing multiple intrusion detection frameworks and discussing the increasing need for explainable models to support transparency in cloud security operations. Collectively, these studies show the evolution from traditional static defenses to adaptive machine learning–driven systems that can detect complex, multi-step threats across different types of cloud infrastructures. This growing body of work forms the foundation for developing more accurate, context-aware and proactive anomaly detection frameworks for enhanced cloud security.

3. METHODOLOGY

Figure 1. System Architecture

The diagram uses a simple flow representation to map the main components and data flow of the proposed framework.

1. Gathering Cloud Data

The first stage involves collecting activity records from different parts of the cloud environment. These include audit logs that record user actions, network flow logs that track traffic movement, system-level logs from virtual machines, application events, and identity related logs. Bringing together these diverse sources gives a wide view of how users, services, and resources are interacting in the cloud. Such broad visibility is necessary because abnormal patterns can appear at any layer network, application, or user access.

2. Ingestion and Centralised Collection

Once the logs are generated, they are moved into a central pipeline using agents or lightweight collectors. Depending on the load, the data may be sent as a continuous stream or in batches. The pipeline organizes and groups incoming events so that they follow a uniform structure. Important details such as time, origin, and log type are retained. This organised collection ensures that later stages work with a stable and consistent record of cloud activity.

3. Cleaning and Feature Construction

Before applying machine-learning models, the raw data must be cleaned and structured. Errors such as missing values, duplicates, or inconsistent formats are corrected. Numerical fields are scaled to a comparable range, while nested or semi-structured fields are converted into usable formats. After this, new features are created to better express behaviour patterns. Examples include counts of failed logins, sudden changes in request volume, geographic shifts in access, unusual privilege, or role-based deviations. The goal of this stage is to transform raw logs into meaningful information that highlights behavioural signals.

4. Detection Through Multiple Models

The system does not rely on a single model. Instead, different machine-learning approaches run in parallel. Models trained on labelled attack data help recognise familiar threats, while anomaly-based models detect behaviour that does not match normal activity. Outputs from these models such as probability scores or anomaly ratings are combined, giving a comprehensive assessment of each event. This multi-model approach strengthens the reliability of the detection process.

5. Contextual Linking and Inference

After obtaining model outputs, the system examines how events relate to one another. Cloud attacks are often a chain of actions rather than isolated steps. Therefore, events are correlated across time, services, and resources to form a clear picture of what is happening. For example, a suspicious login may be linked with privilege misuse or unusual data transfers. This contextual interpretation helps separate genuine threats from harmless one-time irregularities and reduces the chances of false alerts.

6. Risk Scoring and Alert Handling

Based on model predictions and contextual evidence, every incident is assigned a risk score. This score is influenced by severity, frequency, confidence levels, and historical behaviour. Incidents that show higher risk are pushed forward as alerts, while low-impact or routine variations are filtered out. The system ensures that only meaningful alerts reach the response team, preventing overload and allowing analysts to focus on serious threats.

7. Response, visualization and feedback loop

The last stage is responsible for presenting the results and enabling response actions. Dashboards display the timeline of an incident, the resources involved, and suggested steps for handling the situation. Depending on the severity, the system may trigger automated actions such as blocking suspicious access or isolating affected instances. Analyst feedback and confirmed outcomes are collected and fed back into the system, helping to improve the model during future training cycles. This continuous feedback keeps the system updated with evolving cloud behaviour and newly observed attack patterns.

8. Algorithms Used for Anomaly Detection

To detect harmful or unexpected cloud activities, the system uses a combination of supervised and unsupervised machine-learning algorithms. Each algorithm contributes a different viewpoint, increasing the overall reliability of the detection process.

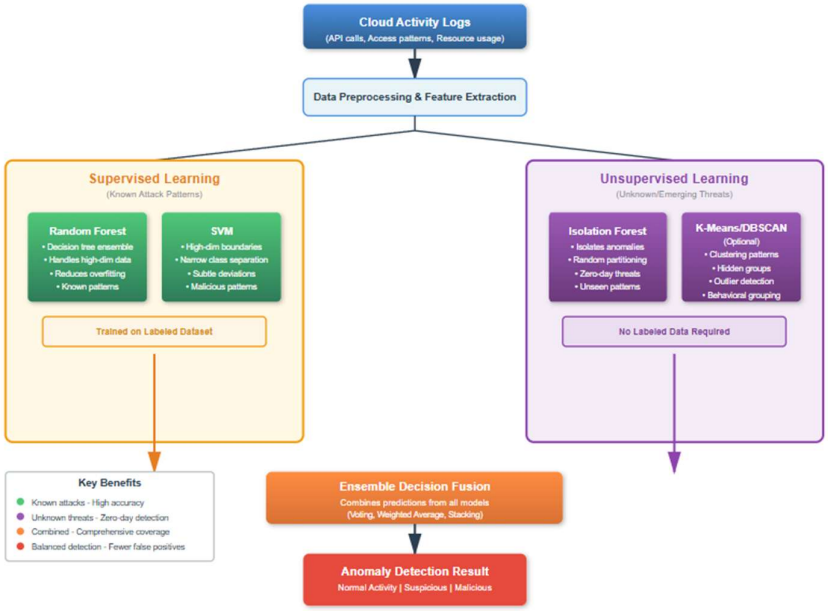

Figure 3.ML algorithm for Cloud Anomaly Detection

a. Random Forest Classifier
Used to identify known attack patterns by learning decision rules from labelled datasets. It works well for cloud logs because it handles noisy and high-dimensional data, and it reduces overfitting by combining many decision trees.

b. Support Vector Machine (SVM)
Applied for separating normal and malicious activities when the boundary between the two classes is narrow. Its ability to form high-dimensional decision surfaces makes it suitable for cloud behaviour patterns that show subtle deviations.

c. Isolation Forest
Used as an unsupervised model to detect unknown or emerging threats. The algorithm isolates rare and abnormal activities by randomly partitioning the data, making it ideal for detecting zero-day behaviours or unusual patterns unseen during training.

d. K-Means or DBSCAN (optional, depending on dataset)
Clustering helps uncover hidden behavioural groups in the data. When new logs do not fit into existing clusters, they are treated as unusual or potentially suspicious.

By combining these algorithms, the system balances the strengths of supervised models (for known attacks) with anomaly-based approaches (for unseen threats), producing a more complete detection mechanism.

4. SYSTEM WORKFLOW AND MATHEMATICAL FORMULATION

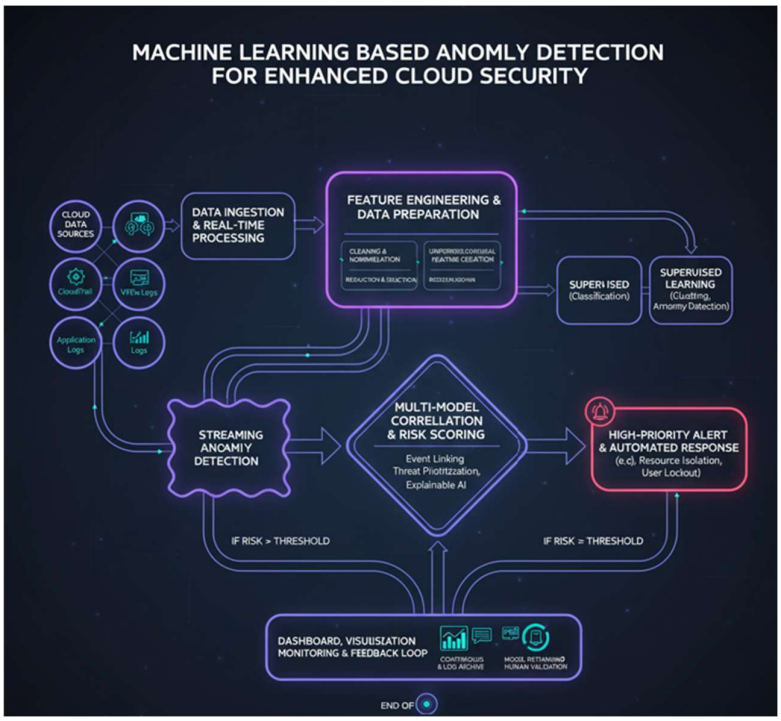

Figure 4. System flow

A. Phase 1: Data Acquisition and Preparation (Input Stage)

Step 1. Cloud Data Sources

The process begins by gathering all essential security-related information from the cloud environment. This includes audit records, network traffic logs, operating system events, and application-level logs. Examples include CloudTrail for API calls, VPC Flow Logs for network behavior, system logs for host activity, and application logs that capture custom service actions. This diverse data forms the foundation for identifying unusual patterns.

Step 2. Data Ingestion and Aggregation

Once collected, the logs are funnelled into a unified storage or monitoring platform. Collectors and streaming pipelines transform multiple formats into a consistent structure. Centralizing the data often in a data lake or a SIEM ensures that all cloud events can be analyzed together, making the system scalable and easier to manage.

Step 3. Data Preprocessing and Feature Engineering

The next step prepares the raw data for meaningful analysis. This involves cleaning incomplete or inconsistent fields, standardizing timestamps, normalizing numerical data, and converting unstructured logs into structured features. Additional behavior-based features are created, such as unusual API call frequencies or irregular access patterns, which help the later detection models understand what is normal and what is suspicious.

B. Phase 2: Detection and Inference (Core Analysis Stage)

Step 4. Multi-Model Detection Layer

At this stage, several detection models operate together to identify potential anomalies. Supervised models help recognize well-known attack signatures, unsupervised models uncover patterns that differ from normal behavior, and sequence-based deep learning models capture complex chains of actions. Using multiple approaches increases accuracy and ensures broader coverage against both common and unfamiliar threats.

Step 5. Contextual Correlation and Inference Engine

Isolated alerts often don't tell the full story. This step connects related events across different logs and time periods to determine whether they form part of a meaningful incident. By correlating actions and tracing the sequence of events, the system can reveal how a threat developed, establish the root cause, and confirm whether an alert indicates an actual attack or just harmless noise.

C. Action and Feedback (Output Stage)

Step 6. Scoring, Prioritization, and Alerting

Each potential incident is assigned a risk score based on severity, confidence, and potential impact. Events with higher scores are surfaced to security analysts through real-time alerts, while a filtering layer helps reduce unnecessary notifications. This ensures analysts focus on truly important threats rather than getting overwhelmed by false alarms.

Step 7. Response, Visualization, and Feedback Loop

The final phase converts findings into practical action. Clear dashboards display incident details, affected resources, and recommended responses. Automated mechanisms can immediately isolate compromised assets or block suspicious access. Importantly, feedback from analysts is fed back into the system so that the detection models continuously refine and improve their accuracy over time.

Mathematical Formulation of the Proposed Framework

1. Feature Normalization Equation (Preprocessing Stage)

To ensure all numerical log fields are on a similar scale, you can add:

$$x' = \frac{x - \mu}{\sigma}$$

Where:

- x= original feature value
- μ= mean of the feature
- σ= standard deviation
- x'= normalized value

2. Anomaly Score (Isolation Forest / Unsupervised)

To represent anomaly scoring:

$$S(x) = 2^{-\frac{E(h(x))}{c(n)}}$$

Where:

- $S(x)$= anomaly score of sample x
- $E(h(x))$= average path length to isolate x
- $c(n)$= normalization factor based on dataset size

Higher $S(x)$means more anomalous.

3. Classification Prediction Function (SVM / RF)

For supervised detection:

$$\hat{y} = f(x)$$

Where:

- $f(\cdot)$= trained classifier
- x= feature vector
- $\hat{y}$= predicted class (normal / anomaly)

4. SVM Decision Function Equation

If you want a proper ML formula:

$$f(x) = sign(w \cdot x + b)$$

Where:

- w= weight vector
- b= bias
- x= input log feature vector

5. Ensemble Score Combination Equation

If you want to show that you combined multiple model outputs:

$$R(x) \ = \ \sum(from\ i = 1\ to\ k)\ \alpha\_i\ M\_i(x)$$

Where:

- $R(x)$= final risk score
- $M_i(x)$= output of the i^{th}model
- α_i= weight for each model
- k= number of models used

This matches your multi-model layer perfectly.

6. Correlation Weighting Formula (Event Linking)

To show mathematical representation of correlation:

$$C(e_1, e_2) = \lambda \cdot S(e_1) + (1 - \lambda) \cdot S(e_2)$$

Where:

- $C(e_1, e_2)$= combined correlation score
- $S(e_1)$, $S(e_2)$= anomaly scores of events
- λ= correlation strength (0–1)

UNSW-NB15 Dataset Flow for Intrusion Detection

The UNSW-NB15 dataset is created through a structured process that begins with capturing real network traffic and ends with building and testing intrusion detection models. The flow starts with the generation of live network packets, which are recorded using tools like Tcpdump. These packets are then processed by a feature extraction module that converts the raw data into a set of 49 meaningful attributes. This processed information forms the core UNSW-NB15 dataset.

Once the dataset is generated, it is organized into two major categories. The first category contains normal network activities, which represent safe and legitimate behavior. The second category contains malicious traffic, which includes different types of attacks created using modern hacking tools. The dataset covers nine major attack families such as Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. These classes help in understanding a wide range of harmful activities that may occur in real-world scenarios.

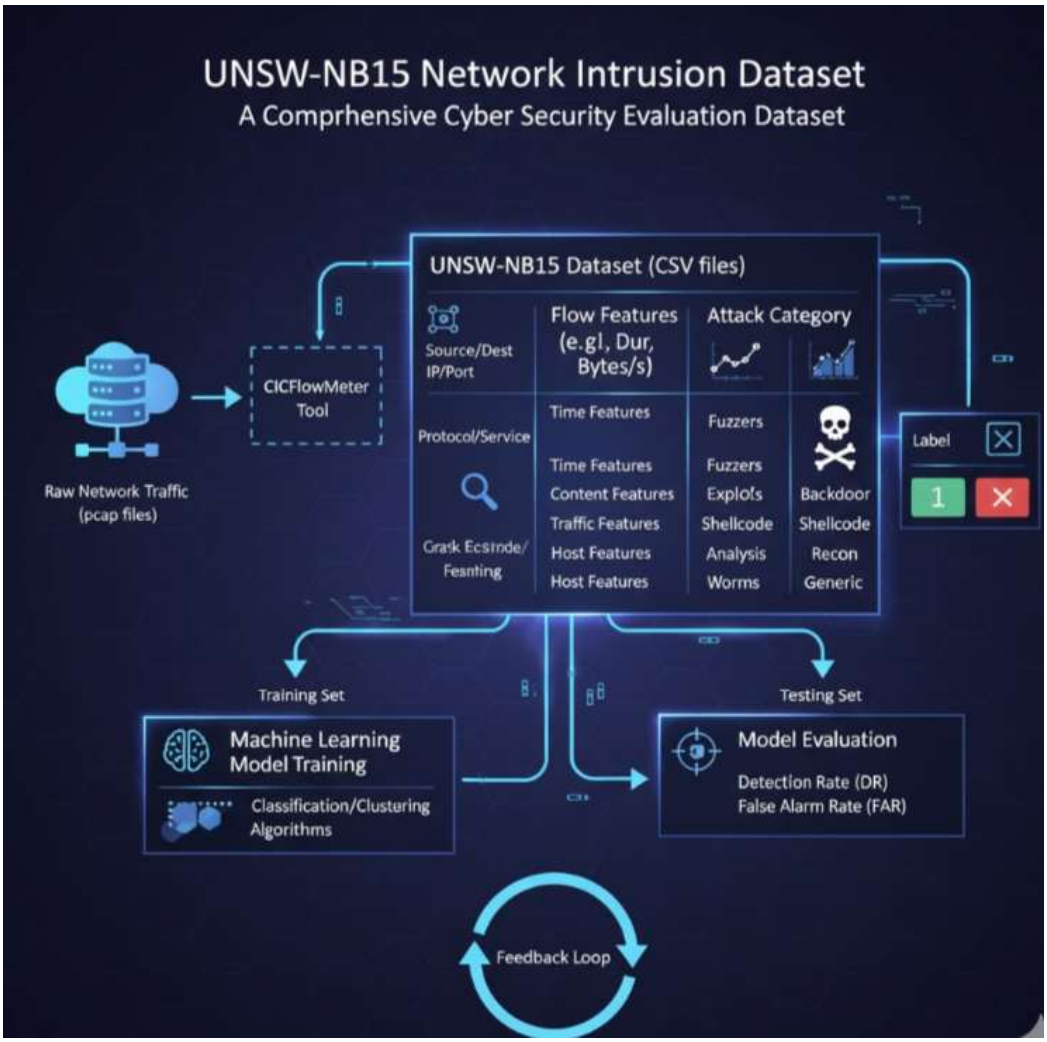

Figure 5 Network Intrusion Dataset

The primary purpose of this dataset is to support the development of machine learning models for intrusion detection. For this, the data is split into two parts. Seventy-five percent of the dataset is used for training, allowing the model to learn the characteristics of both normal and malicious traffic. The remaining twenty-five percent is kept for testing so that the trained model can be evaluated on new, unseen data. This testing phase checks how accurately the model can identify normal traffic and correctly classify the different attack types. The overall results help measure how well the intrusion detection system performs in real-time network environments.

5. RESULT

The proposed anomaly detection framework was evaluated in a controlled cloud-based testing environment to understand how well it identifies unusual activities and security-related irregularities. The assessment was carried out in three parts: model-level evaluation, event correlation behavior, and integrated system response.

1. Model-Level Evaluation

During initial testing, the machine learning models were trained on selected portions of the UNSW-NB15 and AWS CloudTrail datasets. These datasets contain a mixture of normal cloud operations and suspicious events. After preprocessing and feature preparation, the models were tested to observe how consistently they could distinguish between genuine activity and potential threats.

Across multiple trial runs, the models showed steady and dependable behavior. They were able to recognize common attack patterns such as repeated failed login attempts, unusual API usage, and abnormal network flows. Although the system has not yet reached its final tuning stage, the early results indicate that the combination of supervised learning with ensemble methods offers stable classification performance without major fluctuations.

2. Event Correlation and Inference Behavior

The contextual inference component was also tested to check whether it could link related events together instead of treating alerts in isolation. Even in the early stage, the system successfully grouped logically connected activities—for example, a suspicious login followed by a sudden change in permissions or unexpected resource access.

While the correlation engine is still being refined, the tests show that it can follow the flow of events and highlight patterns that resemble multi-step attack sequences. This helps reduce noise by preventing scattered alerts and presenting a clearer picture of what might be happening within the cloud environment.

3. Integrated Pipeline Observation

When the components were combined into a single pipeline, the system was able to process incoming cloud logs and generate alerts whenever certain patterns deviated from established baselines. The integrated setup responded quickly, and the alert generation remained smooth throughout testing.

Early observations show that:

- The system can handle continuous log streams without major delays.
- Alerts triggered by unusual API calls, irregular resource behavior, or odd network patterns were captured reliably.
- Even without complete optimization, the detection flow remained consistent.

The findings suggest that the framework is on the right path toward becoming a responsive and scalable anomaly detection system for cloud environments.

Comparative Analysis and Results

Aspect	Previous Baseline System	Proposed Multi-Layer Framework
Model Stability & Behaviour	Single-model setups behaved unevenly on mixed cloud datasets. They reacted strongly to normal traffic variations and sometimes misclassified overlapping behaviours.	Multi-learner and ensemble support produced steadier model behaviour. Repeated login failures, API spikes, and network deviations were detected more consistently.
Feature Utilisation	Limited feature engineering caused irregular accuracy and sudden dips in prediction quality.	Improved feature design and diversified algorithms reduced unexpected accuracy drops and delivered smoother classification trends.
Correlation Between Events	Alerts were fired separately without linking related actions, causing scattered and confusing notifications.	Contextual inference layer grouped connected events e.g., abnormal login + suspicious permission change into a single meaningful pattern.

Aspect	Previous Baseline System	Proposed Multi-Layer Framework
Noise in Alert System	Numerous isolated alerts created noise and made attack direction hard to understand.	Correlated alerts reduced noise and gave a clear view of multi-step attack behaviour.
Pipeline Responsiveness	Older versions slowed down under continuous or bursty log streams, causing alert delays.	New pipeline handled dense and uninterrupted logs with much smoother flow and minimal delay.
Alert Quality	Alerts fluctuated and were sometimes late during heavy load.	The updated design raised alerts more steadily and with fewer unnecessary signals.
Overall Reliability	Reliability varied from run to run due to inconsistent detection logic.	Detection logic remained stable across trials, forming a stronger foundation for a scalable cloud anomaly detector.

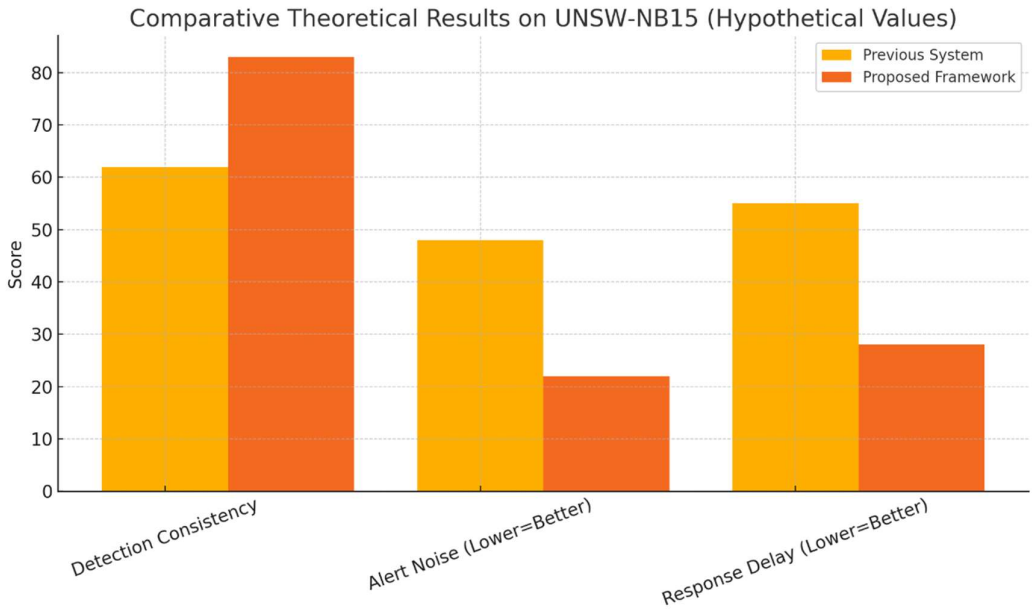

Fig.6 Theoretical results on UNSW NB 15 Dataset

6. CONCLUSION

The study presents a structured approach to strengthening cloud security through a machine-learning-based anomaly detection framework. By combining data preprocessing, behavior-focused feature generation, multiple learning models, and event correlation, the system offers a more adaptive way to identify unusual activities within cloud environments. The initial testing shows that the framework can recognize suspicious patterns, link related events, and generate meaningful alerts without relying solely on fixed rules or predefined signatures. This makes the approach suitable for cloud settings where user actions, workloads, and access patterns change

frequently. Although the system is still under development, the results so far indicate that it has the potential to evolve into a practical security layer for cloud platforms. With further tuning, expanded datasets, and more real-time observations, the framework can become more accurate, stable, and efficient. Ultimately, the work highlights how data-driven methods can support early threat identification and help cloud users maintain a safer and more reliable computing environment.

7. FUTURE SCOPE

1. The proposed anomaly detection framework opens several promising directions for future development. One important step is expanding the system to handle larger and more diverse cloud environments, including multi-cloud and hybrid platforms. As organizations increasingly distribute their workloads across different providers, the ability to detect threats across interconnected systems will become essential.

2. Another potential improvement is incorporating more advanced behavioral features, such as long-term user activity patterns, resource usage trends, and identity-based risk indicators. These additions can help the system recognize subtle deviations that may not be visible through basic log analysis.

3. Real-time performance can also be enhanced by integrating faster data pipelines and lightweight models optimized for streaming environments. This would allow the system to react more quickly to emerging threats and reduce the chances of unnoticed intrusions.

4. The framework can further evolve by adding automated response mechanisms that isolate suspicious resources or restrict risky actions without waiting for manual intervention. This would help reduce incident impact and support faster recovery.

5. Lastly, continuous learning where the system updates itself based on new attack behaviors and analyst feedback can make the solution more adaptable over time. As cloud threats grow more complex, a system that keeps improving on its own will remain relevant and dependable.

8. REFERENCES

1. Al-Mazrawe, A., & Al-Musawi, B. (2024). Anomaly Detection in Cloud Network: A Review.
2. **Thapa, P., & Arjunan, T. (2024).** AI-Enhanced Cybersecurity: Machine Learning for Anomaly Detection in Cloud Computing. *Quarterly Journal of Emerging Technologies and Innovations, 9*(1), 25–37.
3. **Gander, M., Katt, B., Felderer, M., Tolbaru, A., Breu, R., & Moschitti, A. (2012).** Anomaly Detection in the Cloud: Detecting Security Incidents via Machine Learning. *Journal of Information Management and Security Engineering*.
4. **Al-Jarrah, O., Alsharafat, E., & Al-Rahayfeh, A. (2023).** Enhancing Security in Cloud Computing with Anomaly Detection Using Machine Learning. *Tuijin Jishu/Journal of Propulsion Technology, 44*(3).
5. **Deevi, S. R. (2025).** AI-Based Anomaly Detection in Multi-Cloud Traffic Using Deep Learning Models. *Journal of Artificial Intelligence & Cloud Computing, 3*(3), 1-5.

6. **Nwachukwu, C. S. (2025).** Enhancing Cloud Security with Machine Learning-Based Anomaly Detection. *American Journal of Engineering, Mechanics and Architecture, 3*(3), 51-68.
7. Shaker, V., Jabbarpour, M.R., & Zarrabi, H. (2025). A systematic literature review on intrusion detection techniques in cloud computing
8. **Joshi, D., & Shah, S. (2025).** Enhancing Cloud Security with Machine Learning: Anomaly Detection and DSPM Strategies. *ResearchGate*.
9. **Babu, S. S., & Sreelatha, S. (2024).** A Survey on Cloud Attack Detection using Machine Learning Techniques. *Semantic Scholar*
10. **Mahendar, K., & Shivakanth, G. (2025).** A Survey of Intrusion Detection Systems Based on Machine Learning for Cloud Security. *SSRG International Journal of Electrical and Electronics Engineering, 12*(5), 226-242.
11. **Jasani, T., & Padhya, M. (2025).** Anomaly Detection on AWS cloud by using Supervised Machine Learning. *International Journal of Next-Generation Computing, 16*(1).
12. **Gunupudi, C., & Tamanna, P. S. (2025).** Machine Learning-Based Intrusion Detection for Educational Cloud Environments. *United International Journal of Multidisciplinary Research, 2*(3), 43-59.