National College of Ireland

Unsupervised learning for Anomaly and Misconfiguration detection of cloud storage services

MSc Research Project
MSCCYBE_JANO24

Gift Maseno
Student ID: x22242601

School of Computing
National College of Ireland

Supervisor: Raza Mustafa

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Gift Neema Maseno		
Student ID:	X22242601		
Programme	MSCCYBE_JAN024	**Year:**	2025
Module:	Thesis		
Supervisor:	Raza Mustafa		
Submission Due Date:	09 Dec 2025		
Project Title:	Unsupervised learning for anomaly and misconfiguration detection of cloud storage services		
Word Count:	6172	**Page Count**	22

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use another author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: 

Date: 09 Dec 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on a computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

AI Acknowledgement Supplement

BUSINESS RESILIENCE AND INCIDENT RESPONSE

CONTINUOUS ASSESSMENT

Your Name/Student Number	Course	Date
GIFT NEEMA MASENO	MSCCYBE_JAN024_O	09 DEC 2025

This section is a supplement to the main assignment, to be used if AI was used in any capacity in the creation of your assignment; if you have queries about how to do this, please contact your lecturer. For an example of how to fill these sections out, please click here.

AI Acknowledgment

This section acknowledges the AI tools that were utilized in the process of completing this assignment.

Tool Name	Brief Description	Link to tool
Grammarly	Automated grammatical changes	https://app.grammarly.com/

Description of AI Usage

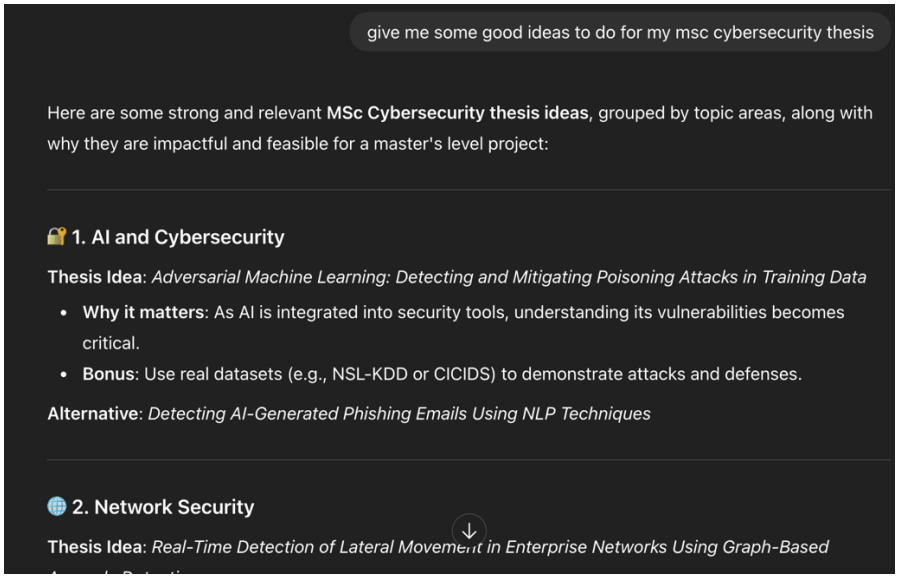
This section provides a more detailed description of how the AI tools were used in the assignment. It includes information about the prompts given to the AI tool, the responses received, and how these responses were utilized or modified in the assignment. **One table should be used for each tool used**.

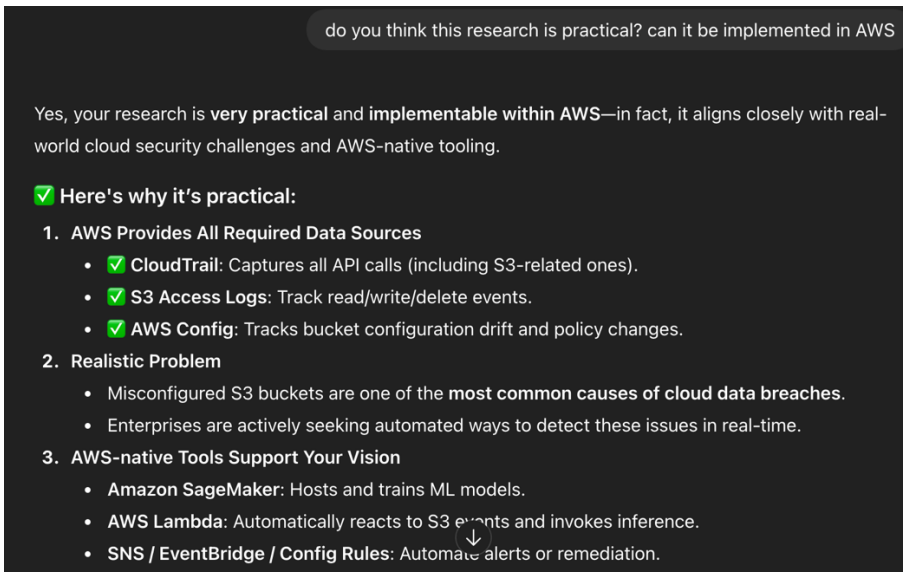
Grammarly	
I used the primary nonpaid version for grammatical changes within the file.	
N/A	N/A
ChatGPT	
I used ChatGPT to get ideas on projects for the implementation of AI&ML in cybersecurity.	

Evidence of AI Usage

This section includes evidence of significant prompts and responses used or generated through the AI tool. It should provide a clear understanding of the extent to which the AI tool was used in the assignment. Evidence may be attached via screenshots or text.

Additional Evidence:





Unsupervised learning for Anomaly and Misconfiguration detection of cloud storage services

Gift Maseno
x22242601

Abstract

Lately, there has been an increasing number of cyber threats arising from cloud service misconfigurations as more organisations adopt cloud technologies. Previous rule-based solutions often fail to adapt to the dynamic nature of cloud environments and noisy data, whilst cloud-native solutions can be challenging to understand and implement, leaving S3 buckets susceptible to various threats such as unauthorised access. With the rise of AI & ML implementations in automated anomaly detection, this study explores the feasibility of applying unsupervised deep learning techniques in identifying misconfigured S3 buckets. This study utilised an open-source dataset from the flAWS challenge environment by Summit Route. Since the dataset was highly imbalanced, a rule-based transformation of data was adopted to simulate both secure and misconfigured fields. The autoencoder model was then trained on secure features to learn normal patterns and detect anomalies via reconstruction error. The model achieved commendable results with an accuracy of 89.6%, precision of 99.7%, recall of 76.9% and f1-score of 0.868, showing the model's capability to detect anomalies with minimal false positives. These results demonstrate the potential of unsupervised models in efficiently detecting and responding to anomalies in cloud environments.

Keywords: *Cloud security, S3 buckets, Misconfigurations, Unsupervised learning*

1. Introduction

Recently, there has been a rapid adoption of cloud computing technologies, which have revolutionised data storage and management by organisations, offering outstanding levels of scalability, flexibility, and cost efficiency. Among its primary offerings, cloud storage services such as AWS S3 buckets, Google Cloud, and Azure Blob storage play a significant role in guaranteeing data accessibility, availability, and management. According to Soares (2025), 96% of companies are expected to adopt public cloud solutions in 2025, with 100 zettabytes of data being stored in the cloud. Nevertheless, data confidentiality, privacy, and compliance are some of the concerns that arise with this implementation, despite the positive contributions of cloud services.

One of the most prevalent and harmful risks posing significant threats to data confidentiality and privacy is S3 bucket misconfigurations, with almost 82% of misconfigurations being attributed to human error (Sentinel One, 2025). Unconstrained Identity and Access Management (IAM) policies, weak or flawed configuration practices, and insufficient log monitoring have led to severe data breaches, for instance, the 2019 Capital One breach, the 2020 Twilio breach, the 2021 Twitch data leak, and the 2025 CodeFinger Ransomware attack (Harpur, 2025). Besides, the lack of standardisation makes it difficult for organisations to implement consistent security policies, which increases the potential for oversight, leading to errors in configuration that are often difficult to identify and remediate.

Current solutions by cloud providers, such as encryption, IAM rules, monitoring dashboards, and system alerts, are largely rule-based and reactive. Besides, previous research, for instance by Continella *et al.* (2018), Van Ede et al. (2022), and Talwar (2024), have explored audit of access control, modelled graphs for IAM policies, and correlation algorithms. However, these methods struggle with scalability, inability to adapt to evolving threats and tactics, coupled with low precision rates leading to high rates of false positives and false negatives. This underscores the need for more adaptive and effective monitoring solutions.

Artificial Intelligence and Machine Learning (AI & ML) methods are a promising solution due to their ability to process large-scale and highly dimensional datasets in real-time cloud environments. Prior studies by Izhikevich *et al.*, (2024) and Lanka, Gupta, and Varol (2024) demonstrate how ML can be used to improve detection accuracy and scalability compared to mainstream approaches. However, there are limited studies specifically focusing on S3 bucket misconfigurations, leaving an opportunity to apply ML-driven anomaly detection to this critical bit of cloud security.

1.1. Research Statement

This research is aimed at assessing whether AI/ML unsupervised learning methods can be used to effectively identify S3 bucket misconfigurations or anomalies in real-time log data for enhanced cloud security monitoring and threat detection.

1.2. Research objectives

This study aims to specifically address S3 bucket misconfigurations by pursuing the following objectives:

i. To assess and review baseline S3 bucket anomaly detection methods and draw conclusions on whether AI/ML-based methods can be used to enhance these techniques.

ii. To design and implement an experimental framework for S3 misconfiguration detection using publicly available datasets.

iii. To evaluate and compare the model's performance and efficacy against conventional and baseline methods using metrics such as accuracy, F1 score, precision, and recall.

By addressing these objectives, this study seeks to explore and demonstrate how AI/ML can strengthen the resilience of cloud storage services by improving detection, mitigation, and response.

1.3. Report structure

The following sections of this paper are organised as follows: Section 2 provides a detailed analysis of various literature that were reviewed relating to common S3 misconfigurations and other studies aimed at addressing the detection of anomalies while leveraging AI/ML. Section 3 covers the methodology for this study, followed by a detailed discussion of the results in Section 4. Section 5 then concludes by highlighting the limitations and opportunities for future work.

2. Related work

2.1. The prevalence of S3 bucket misconfigurations

Cloud misconfigurations in cloud-native storage buckets remain a pain point in cloud security as they pose substantial security risks and data breaches. A team of researchers sought to discover the causes and impact of cloud misconfigurations, analysing over 240,000 S3 buckets (Continella *et al.*, 2018). Their study revealed that many misconfigurations stem from the wrong assignment of permissions due to a lack of knowledge or administrator expertise, hence resulting in data leakage, defacement, and malicious code injection (Continella *et al.*, 2018). The most prevalent S3 misconfiguration is allowing public access, while others include a lack of server-side encryption, misconfigured IAM policies, disabled logging access, and many others (Ozsahan and Worthington, 2022).

Similarly, Guffey and Yanyan (2023) additionally shed more light on this, revealing that S3 buckets can be primarily targeted by attackers through scanning of various IP ranges or

targeted domain names to discover the location of a vulnerable bucket. The vulnerabilities in S3 buckets are often concealed within verbose logs or overlooked system outputs, resulting in unauthorised access, overly permissive IAM roles, exposed network ports, malicious code injection, and data leakage, leading to regulatory violations, reputational damage, and financial losses (Guffey and Yanyan, 2023).

Additionally, Izhikevich *et al.*, (2024) contributed to a better understanding of scanning behaviour used by threat actors leveraging more than a hundred honey buckets, which are intentionally misconfigured S3 buckets with varied names and permissions. Their study revealed that strategies used for scanning vulnerable buckets are methodical and deliberate, targeting big corporations, universities, and governments to find easy compromises for extending malicious attacks like downloading of sensitive files or uploading malicious payload (Izhikevich *et al.*, 2024). Following their study, they developed an algorithm for detecting actors using different Internet Protocol (IP) addresses or Virtual Private Networks (VPNs) for access. These studies demonstrate how easy attackers can scan and abuse misconfigured buckets and the apparent risks arising, like data privacy risks and security breaches.

2.2. AI/ML-driven anomaly detection

Based on the substantial security risks caused by cloud misconfigurations and anomalies, various studies have been conducted to address these. Early attempts of addressing S3 misconfigurations included a CBSAuditor which included a Cloud Security Scoring System (CSSS for continuous monitoring and auditing of expected and actual states of cloud resources to facilitate the detection of misconfigurations, unauthorised changes, and other malicious activities (Torkura *et al.*, 2021). On the other hand, Van Ede *et al.*, (2022) modelled graphs for IAM policies and leveraged the Node2vec embedding algorithm to flag misconfigurations and anomalies. Although their model demonstrated effective and accurate detection between 3.7 to 6.4 times more than traditional rule-based systems, it also indicated slightly lower precision and high rates of false positives and false negatives. Both methods, therefore, appear largely rule-based and static, hence failing to adapt to the rapidly evolving cloud threat landscape.

With the advent of AI and ML, further studies have been conducted to address the static nature of the above conventional methods, offering more scalable and adaptable solutions with high accuracy levels. These techniques can be either supervised, unsupervised, or hybrid approaches.

Supervised ML relies on labelled data, for instance, normal and anomalous traffic, in training the model. This technique can be used in cloud security to train models based on known

threats like unusual login attempts or abnormal network traffic. For instance, Talwar (2024) explored a unified framework for securing cloud-native storage, incorporating anomaly detection algorithms and cloud-native tools for real-time monitoring and proactive identification and remediation of misconfigurations in the multi-cloud environment. This research is significant as it highlights how ML models and correlation algorithms can be integrated with external threat intelligence to identify anomalies in access patterns and configuration settings (Talwar, 2024).

Rachani and Ravish (2025) proposed the XGBoost model for intrusion detection on the cloud, realizing an outstanding performance with an accuracy of 99.81% in differentiating between normal and anomalous traffic. Their research highlights that models can be effective when high-quality datasets are available. However, whilst these studies demonstrated effectiveness in detection, supervised ML methods largely depend on labeled datasets and may fail to adapt to highly dimensional datasets like those in cloud environments, hence could be challenging for such models to identify new or unknown anomalies.

Unsupervised learning methods are a superior approach for anomaly and misconfiguration detection, especially in cloud environments where there is limited labelled data. There are numerous approaches to this, including isolation forests and clustering algorithms like the K-means and DBSCAN that can be leveraged in cloud security for intrusion detection and user behavior analysis (Xu and Yingjie, 2015). Isolation forests randomly partition the datasets and identify anomalies from the data points. Conversely, clustering algorithms measure similarities and dissimilarities within a data structure partition and highlight outliers (Xu and Yingjie, 2015). Since both models learn from a dataset to establish outliers, they are suitable for detecting anomalies and misconfigurations within the cloud environment.

Principal Component Analysis (PCA) is also another approach that significantly diminishes data dimensionality by reducing the complexity of feature space to highlight anomalous behavior, hence improving the efficiency and accuracy of ML models when processing large-scale datasets (Sudharsha *et al.*, 2025). These studies indicate the success of leveraging AI and ML for anomaly detection, reinforcing the proactive implementation for real-time detection of S3 anomalies and misconfigurations.

2.3. Comparison between Supervised and Unsupervised ML methods

Recent studies by Lanka, Gupta, and Varol (2024) and Izhikevich *et al.*, (2024) also offer contrasting yet complementary insights regarding threat detection and S3 bucket misconfigurations. Lanka *et al.* proposed a large language model (LLM) that utilizes honeypot

logs for identifying threats, a testament to how unsupervised ML models can learn and flag anomalies in real-time traffic. On the other hand, Izhikevich *et al.* explored the use of honey buckets, which are intentionally misconfigured S3 buckets, to characterise scanning behaviour used by threat actors. Their study offers insightful findings on methods used by attackers to discover and exploit insecure S3 buckets, hence providing valuable visibility into common Tactics, Techniques, and Procedures (TTPs).

In conclusion, the studies discussed above demonstrate a great potential of AI and ML in detecting configuration anomalies with high accuracy levels. Since there is a limited study specifically focused on real-time detection of anomalies and misconfigurations in storage services like AWS S3 buckets, this research will focus on tailoring an autonomous unsupervised AI/ML pipeline for detecting and responding to anomalous behavior of S3 buckets thereby contributing to a novel and practical solution in a critical cloud security field.

Table 1 below shows a summary of the literature review discussed above.

Author names & year	**Topic**	**Dataset and Models used**	**Findings**	**Limitations**
Continella *et al.* (2018)	There's a Hole in that Bucket! A Large-scale Analysis of Misconfigured S3 Buckets	The researchers utilised over 240,000 honey buckets and utilised a scanner and inspector to check for misconfigurations.	This paper provides an elaborate analysis of popular cloud storage misconfigurations. These misconfigurations such as public read, insecure ACLs, etc., can lead to inadvertent risks such as data leakage, web resource infection, domain name exploitation, etc.	The scanner and inspector introduced in this paper are largely rule-based and can miss on detecting new or unseen threats hence there is opportunity for intelligent and adaptable solutions.
Guffey and Yanyan, (2023)	Cloud Service Misconfigurations: Emerging Threats, Enterprise Data Breaches and Solutions	-	This study explores more current S3 bucket misconfigurations arising from misconfigured IAM roles and open ports. It underscores S3 misconfigurations as one of the leading causes of data breaches such as the Twilio data breach leading to sensitive data leakage and malicious code injection.	The study focuses on exploring common misconfigurations and does not provide a technical solution for S3 bucket misconfigurations.
Lanka, Gupta, and Varol, (2024)	Intelligent threat detection- AI-driven analysis of	Utilised honeypots for collecting log data and developed	This paper demonstrates the success in combining threat intelligence and AI for analyzing	This paper focuses on the TTPs used by attackers to exploit weaknesses in AWS

	honeypot data to counter cyber threats	a K-means LLM for real-time anomaly detection.	large volumes of data for real-time anomaly detection	architecture other than specific S3 misconfiguration detection.
Rachani and Ravish, (2025)	Cloud based IDS-Anomaly detection using machine learning	This paper utilised the NSL-KDD dataset to build both a logistic regression and XGBoost models.	This paper explored a cloud-based intrusion detection system using the logistic and XGBoost model demonstrating high accuracy detection performance when quality datasets are used.	The model is dependent on labelled data affecting its adaptability to novel threats.
Talwar (2024)	Unified Framework for Securing Cloud-Native Storage: Approach for Detecting and Mitigating Multi-Cloud Bucket Misconfigurations	Utilised log data to implement a correlation algorithm for detecting potential anomalies.	This research integrates threat intelligence, ticketing, and ML for multi-cloud security and is significant as it highlights how ML models and correlation algorithms can be integrated with external threat intelligence to identify anomalies in access patterns and configuration settings.	The solution is complex, resource-intensive and is prone to false alarms hence the need for a more lightweight solution that is resource efficient.
Torkura et al. (2021)	Continuous auditing and threat detection in multi-cloud infrastructure	Modelled a CBS Auditor for checking expected and actual states of cloud resources.	Enables continuous monitoring of the states of S3 buckets and alerts on anomalies.	Faces scalability challenges and lacks the mechanisms for reversing undesirable changes.
Van Ede *et al.* (2022)	Detecting Anomalous Misconfigurations in AWS Identity and Access Management Policies	Leveraged labeled misconfigurations to build graph vectors for anomaly detection.	This study shows that graphs can be used for proactive misconfiguration detection considering those that are context specific.	This research is limited to assessing IAM policies and the model shows low precision rates with high false positive and false negative rates.

Table 1: Summary of Literature review

3. Methodology

3.1. Design specification

The key steps followed in this study encompass the development, implementation, and evaluation of an autonomous framework for detecting S3 bucket misconfigurations or

anomalies using AI/ML in line with the design science research methodology (DSRM). First, the review of relevant publicly available datasets for training the model was completed.

Once the suitable dataset was selected, the iterative process of training the model began, extracting pertinent features from the dataset. The development was conducted using Python in the Google Colab environment, as it is a flexible open-source platform providing computational efficiency and seamless integration with TensorFlow and Scikit-learn libraries.

Finally, the performance of the model was evaluated in line with key metrics such as precision, recall, accuracy, and reconstruction error.

Figure 1 below illustrates the proposed architecture of the model.

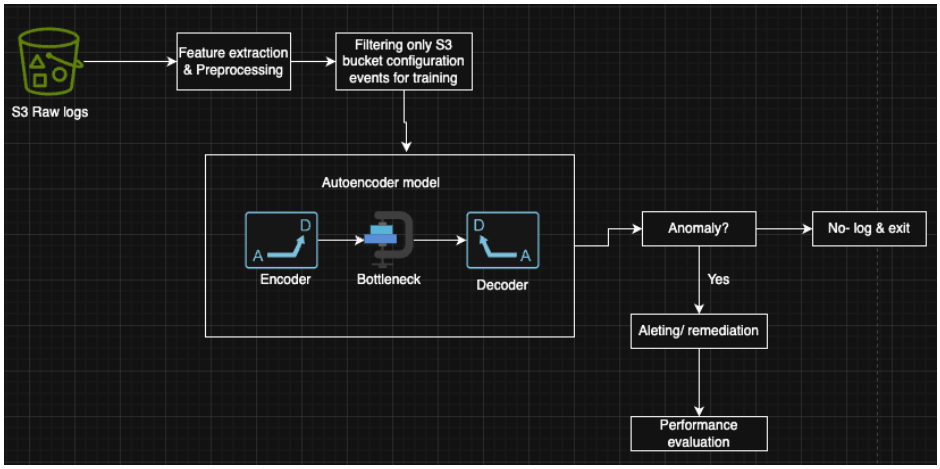


Figure 1: System architecture of the proposed autoencoder model

3.2. Proposed model

This study explored the development of an autoencoder pipeline in detecting S3 misconfigurations. An autoencoder is an unsupervised neural network that efficiently learns codings of unlabeled data by compressing or encoding inputs into a lower-dimensional latent space and reconstructing or decoding the original input from the compressed form (Bergmann and Stryker, 2023). This capability enables autoencoders to discover latent variables, playing a crucial role in data denoising, dimensionality reduction, generative modelling, and anomaly detection, hence why it was leveraged in this study (Bergmann and Stryker, 2023).

During training, the model reduces the reconstruction error between the input and output variables. Misconfigured events and anomalous configurations are represented by higher reconstruction errors because of their deviation from normal behavior, which makes autoencoders useful in detecting cloud anomalies like S3 misconfigurations (Bergmann and Stryker, 2023).

The structural elements of an autoencoder are:

- An **encoder** that encodes compressed input data, extracts important features, and reduces dimensionality.
- The **bottleneck or code**, which is the compressed form of the input, referred to as the latent space that is fed to the decoder.
- A **decoder** that decompresses or decodes and reconstructs the data back to its original form

Figure 2 below shows the architecture of the autoencoder.

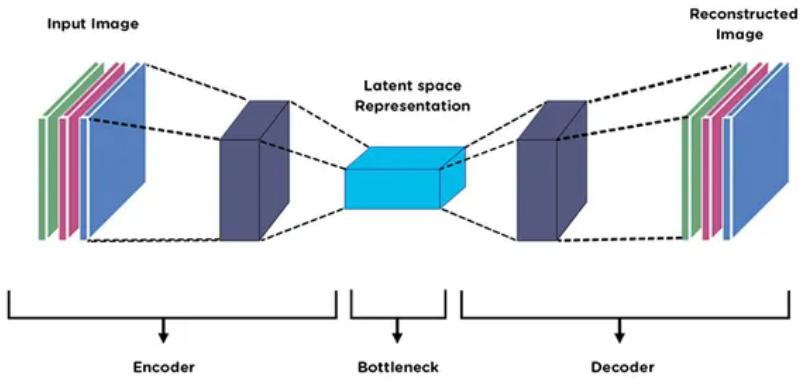


Figure 2: Architecture of an autoencoder

3.3. Dataset selection and justification

The main criterion for selecting the dataset to be used for this study was the ability to reflect the latest real-world cloud resource access patterns. Two publicly accessible datasets were evaluated, sourced from GitHub and Summit Route, respectively.

The first[1] dataset was `invictus-ir/aws_dataset` from GitHub, simulating AWS CloudTrail attack events. This dataset, however, appeared synthetic and was specific in modelling adversarial behavior rather than providing a holistic representation of cloud resource interactions, including S3 buckets, and hence did not meet the goals of this research in having a dataset that mimics real-world variance.

The second[2] dataset was sourced from the Summit route and consisted of over 1.9 million event entries of raw and anonymized AWS CloudTrail logs generated from Scott Piper's flAWS challenge environment. This dataset was chosen as it included useful S3 bucket features and configuration patterns such as public buckets, encryption features, and IAM

1. Bert (2023) 'Invictus Incident Response'. GitHub. https://github.com/invictus-ir/aws_dataset.

2. Piper, S. (2020) 'Public dataset of Cloudtrail logs from flaws.cloud'. SummitRoute. https://summitroute.com/blog/2020/10/09/public_dataset_of_cloudtrail_logs_from_flaws_cloud/.

privileges, which are significant for this study. Therefore, although it was highly unstructured because of the raw logs and required extensive preprocessing and feature extraction, it was ultimately selected given its richness, scale, and focus on storage-related activity. It is highly relevant for the objectives of this research as it simulates realistic access patterns and misconfigurations, together with mirroring the attack vectors of S3 buckets.

3.4. Feature extraction and pre-processing

A structured pre-processing pipeline was employed before model training to prepare the dataset for the machine learning model. The *JSON.GZ* data files were extracted from the *.tar* Summit route archive and parsed into a structured format before combining them into a pandas data frame. This process included flattening the nested structures in each JSON log file to enable exploratory data analysis (EDA). An initial inspection was done to display the key structure and data format of the sampled records, as shown below.

Figure 3: Visualization of files from the .tar archive

3.5. Feature engineering

Due to the complexity of the dataset, eight pertinent features in Table 2 below were chosen for assessment in this study as they are highlighted to be the leading misconfigurations experienced in the AWS architecture (TrendMicro, 2021). The categorical features were encoded into a binary representation, with misconfigured buckets represented as 1 and secure ones as 0. This was an essential step as most machine learning models cannot process raw categorical values (Geeks for Geeks, 2025). Buckets with inconclusive data were retained, and the missing values were computed and handled by imputation (NaN → 0), assuming the absence of information as the default configuration.

Feature	**Description**
`Publicly_accessible`	Denotes the presence of publicly accessible S3 buckets.

`Policy_public`	Checks for buckets allowing public access via "Principal": "*"
`Logging_enabled`	Checks whether logging is enabled.
`Public_access_block_enabled`	Checks if AWS Public Access Block is enabled
`Acl_insecure`	Identifies misconfigured ACLs granting public read/write permissions.
`Encryption_enabled`	Checks whether server-side encryption (SSE) is enabled.
`Versioning_enabled`	Checks whether versioning is enabled to prevent overwriting of data.
`Cors_unrestricted`	Identifies unrestricted configurations that could allow requests regardless of the origin.

Table 2: S3-related record features used for model training

3.6. Data Labeling and Balancing

The generated dataset was highly imbalanced, with a significant percentage of the buckets having unknown states, whereas a negligible percentage contained secure and misconfigured buckets, as shown below. A bucket is considered secure when none of the highlighted configuration attributes render the bucket susceptible to potential unauthorized access or data leakage, while a misconfigured bucket contains one or more attributes that are either disabled, missing, or overly permissive. This low representation reflects the natural sparsity of configuration-change events in operational CloudTrail logs.

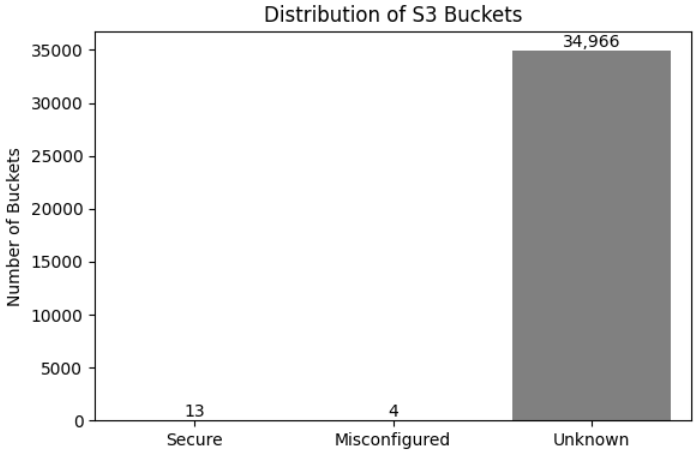


Figure 4: Distribution of S3 buckets

Therefore, to balance the dataset for effective evaluation and address the scarcity of misconfigured fields, a rule-based data transformation approach was utilised. This approach

was dependent on domain knowledge of S3 bucket misconfigurations rather than stochastic or interpolation-based augmentation. Starting with the secure subset of buckets, the eight configuration attributes selected for this study were programmatically transformed to a binary of 40, and a seed sample of 5000 was used for balancing and reproducibility.

This process is significant in preserving validity and interpretability by ensuring that each security configuration corresponds to a plausible AWS scenario. Other alternative approaches, such as SMOTE, GAN-based generation, and random noise injection, were deemed unsuitable as they can produce non-discrete feature values with no semantic meaning. Rule-based transformation was, therefore, ultimately selected because it preserves the logical and semantic integrity of S3 configuration data while generating scientifically valid, interpretable synthetic misconfigurations for model evaluation. As a result, a balanced dataset was generated with approximately 50% of secure records and 50% of misconfigured buckets, as shown in Figure 5 below, while Figure 6 shows the distribution of each misconfiguration type. This process was significant in ensuring a comprehensive learning process of the model by reducing the risk of bias and underfitting.

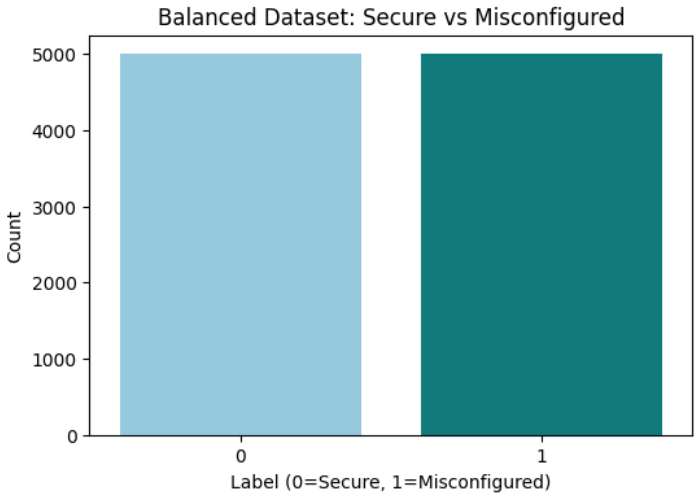


Figure 5: Balanced dataset of secure and misconfigured buckets

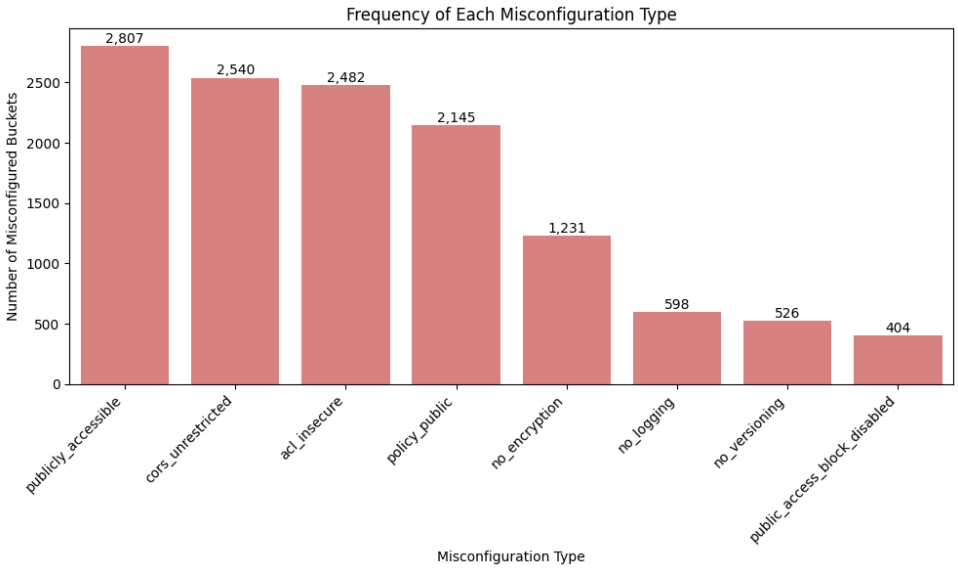


Figure 6: Frequency of each misconfiguration type

3.7. Train-test-split and Scaling.

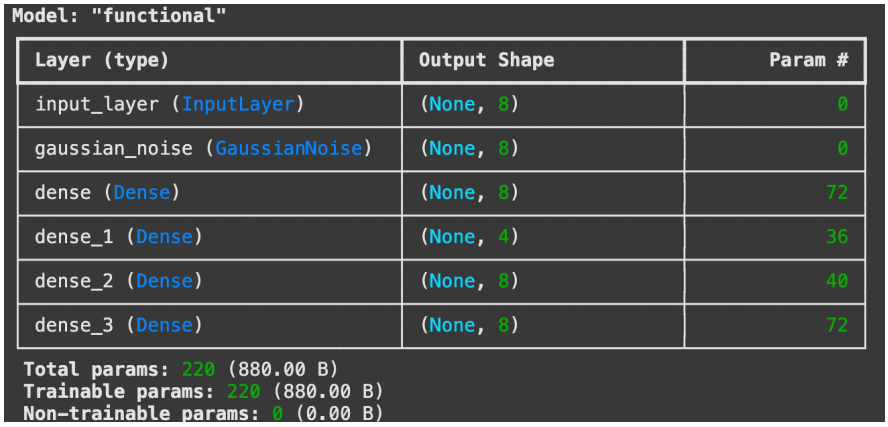
After completion of pre-processing, the dataset was portioned into a training set and test set with a 70:30 ratio, where 70% of the samples were allocated for model training, while 30% were withheld for model evaluation. The stratify parameter was initialized during the splitting process to balance of class distribution between normal and abnormal configurations. Subsequently, standardization was performed utilizing the *StandardScaler* from the Scikit library for neural convergence, ensuring that features with large numeric ranges did not dominate the learning process of the model. This process was significant in enforcing data integrity, improving data quality, and enforcing consistency so that the autoencoder can achieve good performance and generate reliable results.

3.8. Model training

S3-specific trainable sample parameters were selected for model training to improve computational efficiency and expressive power. An autoencoder pipeline was established with the following architecture using the TensorFlow and Keras libraries.

- An input layer made up of eight neurons to match the feature set.
- An encoder made up of two dense layers reducing input dimensionality from 8 → 4 → 3 neurons to capture the compressed latent of normal configurations.
- A Gaussian noise layer of 0.05 is applied on the input layer to increase robustness.
- A decoder that reconstructs the input to its original form (3 → 4 → 8).
- An output layer for linear activation for the reconstruction of standardized outputs.

Figure 7 below demonstrates the definition of the model.



Model: "functional"

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 8)	0
gaussian_noise (GaussianNoise)	(None, 8)	0
dense (Dense)	(None, 8)	72
dense_1 (Dense)	(None, 4)	36
dense_2 (Dense)	(None, 8)	40
dense_3 (Dense)	(None, 8)	72

Total params: 220 (880.00 B)
Trainable params: 220 (880.00 B)
Non-trainable params: 0 (0.00 B)

Figure 7: Definition of input and output layers for the autoencoder

The model can use reconstruction error analysis to identify deviations from normal configurations by learning to reconstruct input characteristics from their compressed forms. To reduce reconstruction errors between the input and output and optimize the model, the Adam optimizer was included in compilation with a learning rate of 0.001 and mean squared error (MSE) for the loss function. The model was trained on secure data using 80 epochs to facilitate iterative improvement of the model, and an early stopping rate of `(patience=8)` was included to prevent overfitting. The model's capacity was purposely kept low to ensure that it could not easily reconstruct unseen misconfigurations.

3.9. Ethical considerations

All phases of this study adhere to fundamental ethical guidelines for information security implementation:

- **Data privacy and Anonymisation:** the flAWS dataset contains pseudonymised CloudTrail logs with no personal data. All residual identifiers, like IP addresses, were also hashed before analysis to prevent re-identification.
- **Compliance with the Institution:** An attestation with the university was done for approval of this study's methodology, ensuring compliance with the General Data Protection Regulation (GDPR) and other data protection regulations.
- **Responsible disclosure:** The flAWS dataset used in this study contains synthetic misconfigurations that were generated in a sandbox environment; hence, it does not expose any live AWS credentials or S3 buckets in production.
- **Transparency and Reproducibility:** all experimental code and datasets that are not sensitive datasets will be released under an open-source licence for peer evaluation, while redacting any information that would identify real systems.

4. Discussion of results

4.1. Overview

This section details the results obtained from the implementation of the proposed AI/ML framework for detecting S3 bucket misconfigurations. The model was evaluated based on its performance, feature behaviour, and interpretability.

An autoencoder pipeline was established in the Google Colab environment as described in the previous section using the FlAWS CloudTrail dataset. The fundamental goal was to implement an effective unsupervised learning AI/ML pipeline that would enable automated real-time detection of S3 misconfigurations such as publicly accessible buckets, lack of encryption, insecure ACLs, missing logging or versioning, and unrestricted CORS.

4.2. Model evaluation

Evaluation of the model was conducted to measure the accuracy and effectiveness in detecting anomalous configurations by reconstructing inputs from the test set and computing the reconstruction error. A cut-off point within the 90^{th} and 99^{th} percentiles of reconstruction errors from typical samples was established, with higher errors being flagged as outliers, suggesting possible S3 misconfigurations.

A comprehensive assessment of the performance of the model was then conducted based on the following metrics:

- **Accuracy**: a measure of correct predictions out of all predictions made by the model.
- **Precision** is the percentage of true positive predictions relative to all positive predictions made by the model, signifying the model's precision in correctly identifying positive instances.
- **Recall**, also known as sensitivity, represents the model's ability to accurately identify positive instances out of all actual positive instances, demonstrating a strong capability to capture true positives.
- **F1 Score**: is a composite metric that balances precision and recall, providing an overall measure of model performance.
- **Reconstruction error distribution plot**: provides visual separation between normal and anomalous configurations.
- **Confusion matrix** for visualization of distribution in classifications.

4.3. Model performance

Multiple tests were conducted to test the performance of the model. Tests were conducted with 50 and 80 epochs, and it was observed that there was convergence at about 40 epochs, after which both the training and validation loss stabilised. Therefore, we settled on training using 80 epochs of normalised input features for establishing a balance between flexibility and computational efficiency. The Mean Squared Error (MSE) steadily declined from 0.003-0.005, demonstrating the model's ability to gradually learn the structure of normal configuration patterns. The Gaussian-noise layer helped to prevent overfitting by enabling generalisation of normal behaviour rather than memorisation of specific patterns.

When reconstruction error was computed, normal buckets appeared to be concentrated near zero, whereas the synthetically generated misconfigurations produced higher errors seen further along to the right, as shown below.

Figure 8: Reconstruction error

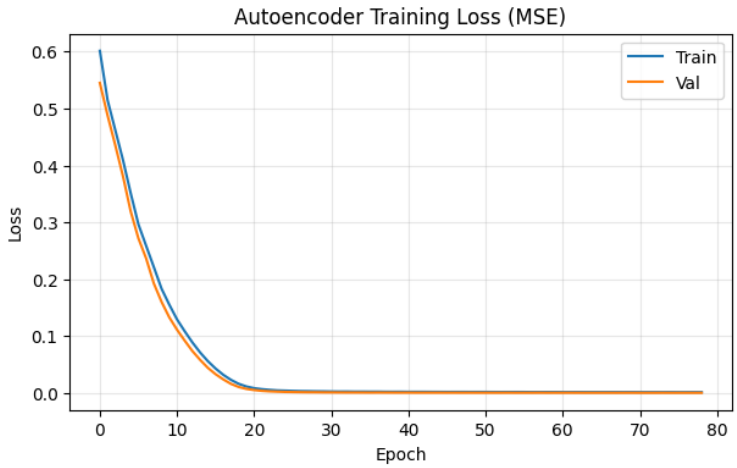


Figure 9: Mean Squared Error validation loss

4.4. Performance metrics and discussion of results

When the first experimental results were done, the model produced a perfect 100% in accuracy, precision, and recall. However, while the results appeared exceptional, they raised a question about whether this was a realistic performance of the model. Further analysis suggested that this performance could have been a result of data leakage or overfitting with overlapping samples between training and test sets, or the synthetic dataset was too distinct.

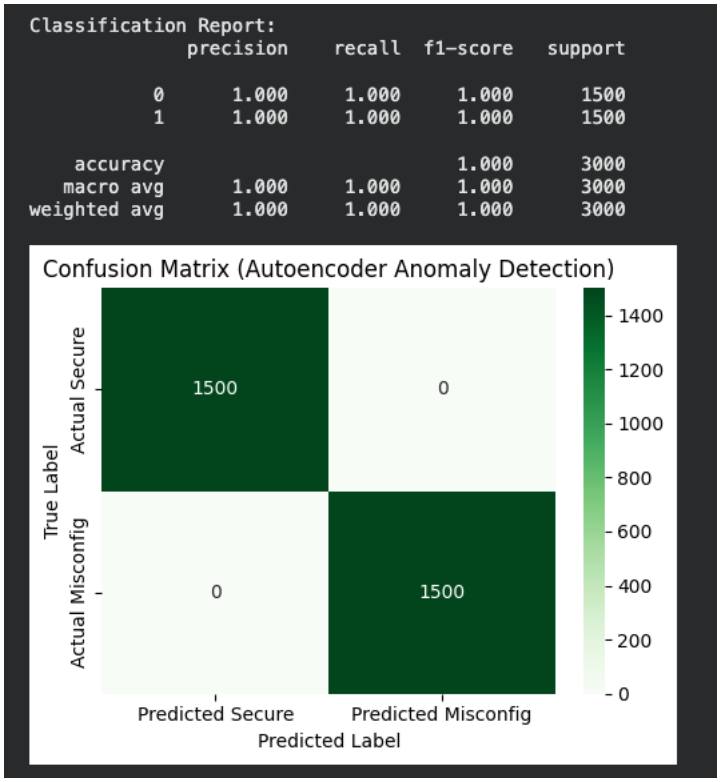


Figure 10: Initial model performance metrics

Therefore, further analysis was done implementing a clean data split with protective safeguards for limiting data leakage, as seen in Table 3 below. The autoencoder demonstrated a more realistic performance, achieving an overall accuracy rate of 89.%, correctly classifying 3031 out of 3382 test samples. The model yielded an exceptionally high precision of 99.7% for the misconfigurations class, demonstrating its ability to correctly predict misconfigurations that render buckets as insecure. This is a desirable property when designing automated cloud security systems, as it lowers the likelihood of false warnings leading to increased trust by analysts. However, while most misconfigurations were successfully detected, the recall of 76.9% suggests the inability of the model to detect some subtle or partial anomalies. This can be attributed to the conservative bias of unsupervised autoencoders, which prioritize minimizing false positives over maximizing sensitivity. Conversely, the F1 score of 0.868 demonstrates a well-balanced trade-off between precision and recall, showcasing the model's

capability to capture essential characteristics of secure buckets while remaining sensitive to deviations that could highlight misconfigurations. The model also had a significantly low positive rate of 0.001003, which shows that the model can identify deviations without overfitting to noise.

Metric	Value	Interpretation
Accuracy	89.6%	The cumulative predictions show good classification behavior.
Precision	99.7%	The model is very reliable with negligible rates of false alarms.
Recall	76.9%	The model failed to detect subtle misconfigurations.
F1 score	0.868	Good balance between sensitivity and specificity in avoiding false alarms.
False Positive Rate	0.001%	Meets the <1% threshold.

Table 3: Performance metrics

The confusion matrix below provides a visual representation of the performance of the model, indicating that only 4 false positives and 347 false negatives occurred.

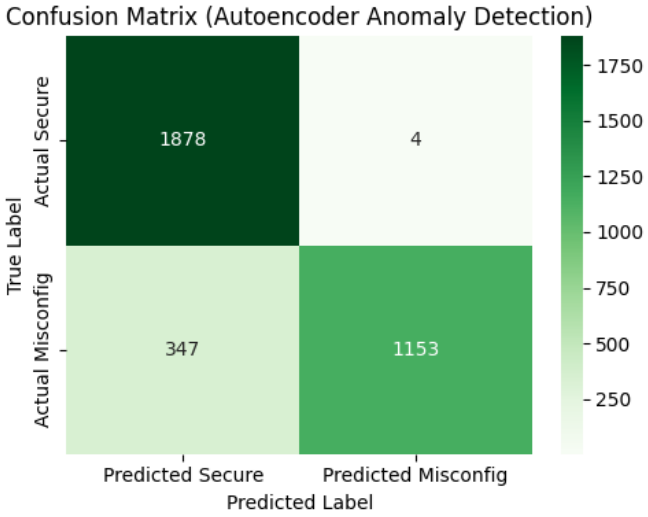


Figure 11: Confusion matrix

4.5. Summary

In summary, these results demonstrate the model's ability to effectively model S3 configuration patterns without relying on labelled datasets or explicit supervision, underscoring its robustness and practical implementation in addressing misconfigurations.

The ability to learn from normal behavioural patterns and detect anomalies solely on reconstruction error makes it suitable for implementation in real-time monitoring scenarios for

large-scale cloud environments. However, the false negatives indicate that some misconfigurations closely correlate to secure configurations in the statistical representation, suggesting that future research should explore hybrid techniques such as ensemble learning to fine-tune the model's sensitivity to subtle anomalies. Besides, although the results were obtained using a controlled synthetic dataset, they provide strong foundational evidence about the feasibility of using unsupervised learning methods to identify and mitigate configuration-based security vulnerabilities in cloud environments.

5. Limitations and Future Work

Whilst the developed model demonstrated high reliability and capability in distinguishing between anomalous and normal S3 configurations, a few limitations were identified as opportunities for further enhancement.

Firstly, this research was **limited to evaluating AWS S3 bucket misconfigurations,** as AWS services are widely adopted. However, the methodology can be implemented in evaluating other forms of anomalies and services from other cloud providers like Microsoft Azure, Google Cloud Services, etc.

Additionally, the recall of 76.9% is an indication that the model can potentially fail to detect subtle anomalies. This could be caused by close correlation of insecure features to normal feature states making them appear as normal, thereby eluding the anomaly detection boundary learned by the model.

Another limitation experienced was **the synthetic nature of the dataset used**. The flAWS dataset was highly imbalanced, which necessitated rule-based generation of synthetic samples for model training and validation. Therefore, the resulting dataset may not fully represent the practical complexity, interdependence, and noise seen in real-world AWS S3 environments. Consequently, whilst the current model performs well with structured and clearly separable data, enhanced validation is required leveraging more realistic configuration logs that have diversity in configuration settings and anomaly patterns.

Furthermore, the autoencoder model proved to be **resource-intensive and computationally intensive to build,** especially when handling large datasets in a cloud environment. These complexity challenges were observed in data extraction and pre-processing, which indicate that this model would not be suitable in resource-constrained environments.

Future work should, therefore, **focus on expanding the dataset by integrating real-world AWS CloudTrail or configuration data** that represents practical implementation scenarios. This would enhance diversity by having various representative values that allow the model to learn from naturally occurring configuration drift. To improve the reliability of the model, the model architecture can be enhanced by exploring variational autoencoders (VAEs) or tabular generative adversarial networks (GANs) to make the synthetic data more realistic and address current distribution gaps. A hybrid framework combining the autoencoder with supervised fine-tuning or ensemble learning could also be considered to improve recall and interpretability of the model.

This work did not achieve integrating the model with the AWS architecture. Therefore, **practical integration with AWS cloud-native services** would also be a significant extension of this work to enable a fully automated and scalable solution. For instance, Amazon SageMaker could be leveraged for dynamic training and deployment of the model, enabling continuous learning and adaptation of the model to the ever-changing cloud environment. AWS Lambda can then be used for automated detection to allow the autoencoder inference to execute automatically in response to configuration changes or newly ingested CloudTrail events. This serverless approach would lead to a lightweight and cost-efficient pipeline. In addition, Amazon Simple Notification Service (SNS) can be leveraged to provide alerting and. notification capabilities in the event of a misconfiguration being detected to improve incident response time, facilitate prompt remediation actions. Together with CloudWatch, these features can be used for continuous monitoring, auditing, and reporting of misconfiguration trends across large-scale environments.

6. Conclusion

This work assessed the effectiveness of unsupervised machine learning techniques in detecting S3 bucket configuration anomalies in real-time log data for enhanced threat detection and response. It employed a denoising autoencoder pipeline which achieved satisfactory results with system accuracy of 89.6%, precision of 99.7% and a negligible false positive rate of 0.001%, demonstrating very high reliability with minimal dependence on labelled datasets.

This approach demonstrates better performance compared to the previous rule-based and supervised methods discussed in the literature, as it is more adaptable and scalable. In addition, this framework offers a realistic foundation for integration with AWS cloud-native solutions such as AWS Lambda for automated remediation, SageMaker for model orchestration, and SNS for generating alerts to enable a fully autonomous pipeline.

Therefore, this work has provided a novel, practical and scalable solution to a critical cloud security field. The findings would be beneficial in bridging the gap between theoretical AI research and operational cloud defence, offering unsupervised learning as a promising solution for cloud security automation.

References

Bergmann, D. and Stryker, C. (2023) 'What is an Autoencoder?', *IBM*, 28 November. https://www.ibm.com/think/topics/autoencoder.

Continella, A. *et al.* (2018) 'There's a Hole in that Bucket!: A Large-scale Analysis of Misconfigured S3 Buckets', in. *ACSAC '18: 2018 Annual Computer Security Applications Conference*, PR, San Juan, USA: ResearchGate, pp. 702–711. https://doi.org/10.1145/3274694.3274736.

Geeks for Geeks (2025) 'Label Encoding in Python', *Geeks for Geeks*, 12 February. https://www.geeksforgeeks.org/ml-label-encoding-of-datasets-in-python/ (Accessed: 26 April 2025).

Guffey, J. and Yanyan, L. (2023) 'Cloud Service Misconfigurations: Emerging Threats, Enterprise Data Breaches and Solutions', in. *2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, N.V, USA: IEEE, pp. 0806–0812. https://doi.org/10.1109/CCWC57344.2023.10099296%257D.

Harpur, R. (2025) 'AWS Data Breach: Lessons from 4 High Profile Breaches', *Technology*, 29 January. https://www.blackfog.com/aws-data-breach/.

Izhikevich, K. *et al.* (2024) 'Using honeybuckets to characterize cloud storage scanning in the wild', in. *2024 IEEE 9th European Symposium on Security and Privacy (EuroS&P)*, Vienna, Austria: IEEE, pp. 95–113. https://doi.org/10.1109/EuroSP60621.2024.00014.

Lanka, P., Gupta, K. and Varol, C. (2024) 'Intelligent threat detection- AI-driven analysis of honeypot data to counter cyber threats', *Data Security and Data Analytics in Cloud Computing*, 13, p. 2465. https://doi.org/10.3390/electronics13132465.

Ozsahan, H. and Worthington, D. (2022) 'Top 5 Amazon S3 bucket misconfigurations and how to monitor them', *JumpCloud*, 25 May. https://jumpcloud.com/blog/amazon-s3-bucket-misconfigurations.

Piper, S. (2020) 'Public dataset of Cloudtrail logs from flaws.cloud'. SummitRoute. https://summitroute.com/blog/2020/10/09/public_dataset_of_cloudtrail_logs_from_flaws_cloud/.

Rachani, R.M. and Ravish, R. (2025) 'Cloud-based IDS-Anomaly detection using machine learning', in. *2025 1st International Conference on AIML-Applications for Engineering & Technology (ICAET)*, Pune, India: IEEE, pp. 1–6. https://doi.org/10.1109/ICAET63349.2025.10932232.

Sentinel One (2025) '50+ Cloud Security Statistics in 2025', *Sentinel One*, 3 April.

Soares, M. (2025) 'Cloud Computing Stats 2025', *Nextwork*, 11 February. https://www.nextwork.org/blog/cloud-computing-stats-2025 (Accessed: 6 May 2025).

Sudharsha, S.C. *et al.* (2025) 'Adaptive network intrusion detection using fine-tuned random forest with PCA', in. *2025 International Conference on Advanced Computing Technologies (ICoACT)*, Sivalasi, India: IEEE, pp. 1–6. https://doi.org/10.1109/ICoACT63339.2025.11004662.

Talwar, S. (2024) ‘Unified Framework for Securing Cloud-Native Storage: Approach for Detecting and Mitigating Multi-Cloud Bucket Misconfigurations’, *Computer Fraud & Security*, 2024, pp. 341–348.

Torkura, K.A. *et al.* (2021) ‘Continuous auditing and threat detection in multi-cloud infrastructure’, *Computers & Security*, 102. https://doi.org/10.1016/j.cose.2020.102124.

TrendMicro (2021) ‘The most common cloud misconfigurations that could lead to security breaches, *Trend Micro*, 25 October. https://www.trendmicro.com/vinfo/ie/security/news/virtualization-and-cloud/the-most-common-cloud-misconfigurations-that-could-lead-to-security-breaches.

Van Ede, T. *et al.* (2022) ‘Detecting Anomalous Misconfigurations in AWS Identity and Access Management Policies’, in. *CCSW’22: Proceedings of the 2022 on Cloud Computing Security Workshop*, Los Angeles, CA, USA: Association for Computing Machinery, pp. 63–74. https://doi.org/10.1145/3560810.356426.

Xu, D. and Yingjie, T. (2015) ‘A comprehensive survey of clustering algorithms’, *Annals of data science*, 2, pp. 165–193. https://doi.org/10.1007/s40745-015-0040-1.