







# Efficient Post-Processing of Intrusion Detection Alerts Using Data Mining and Clustering

Kalu Gamage Kavindu Induwara Kumarasinghe<sup>1</sup><sup>a</sup>,  
Ilangan Pakshage Madhawi Pathum Kumarsiri<sup>1</sup><sup>b</sup>,  
Harsha Sandaruwan Gardiyawasam Pussewalage<sup>2</sup><sup>c</sup>,  
Kapuruka Abarana Gedara Thihara Vilochana Kumarasinghe<sup>1</sup><sup>d</sup>,  
Kushan Sudheera Kalupahana Liyanage<sup>1</sup><sup>e</sup>, Yahani Pinsara Manawadu<sup>1</sup><sup>f</sup> and Haran Mamankaran<sup>3</sup>

<sup>1</sup>University of Ruhuna, Sri Lanka

<sup>2</sup>University of Agder, Norway

<sup>3</sup>Sysco Labs, Sri Lanka

**Keywords:** Clustering, Cyber Security, Data Mining, Intrusion Detection System, Unsupervised Learning.


**Abstract:** Network Intrusion Detection Systems (IDS) have evolved significantly over the past two decades to address the growing complexity of network infrastructures and the increasing volume of cyber threats. However, traditional IDS approaches either rely on predefined signatures, which fail to detect zero-day attacks, or use anomaly detection models that suffer from high false alarm rates, overwhelming security analysts with excessive alerts. This paper proposes a data mining and adaptive clustering-based unsupervised approach to efficiently process IDS-generated network alerts, reducing false positives and enhancing threat detection. Relevant alert features are extracted, and advanced data mining techniques are applied to identify frequent patterns, reducing false alerts. Clustering similar patterns further groups alerts from related attacks, thereby reducing the workload of security analysts. This allows analysts to gain a high-level understanding of intrusions without manually reviewing vast numbers of alerts. The approach further enhances intrusion detection accuracy and provides actionable insights through alert correlation. The experimental results demonstrate significant improvements in detecting various cyber threats, including DDoS, Botnets, Port-scans, and more.


## 1 INTRODUCTION


In the past two decades, network intrusion detection has shown remarkable growth. Due to networks becoming increasingly complex, it has become necessary to develop effective approaches based on learning architectures to solve operating problems such as high volume of intrusion alerts and false alarms. However, anomaly detection has become challenging because of networks' increasing structural complexity, which requires many variables to be monitored (Ramírez et al., 2023). Attackers can exploit vul-


nerabilities in security systems, such as insufficient detection capabilities, to infiltrate networks. Then they gain access to sensitive information, potentially causing the system to malfunction and breach data confidentiality (Abdulganiyu et al., 2023). Therefore, Intrusion Detection Systems (IDSs) are crucial cyber-security tools which monitor and analyse network traffic and detect and report malicious activities to counter potential attacks (Abdulganiyu et al., 2023).


Advanced IDS systems are usually rely on anomaly detection and machine learning techniques. Most research on anomaly detection has focused on improving detection accuracy, especially as these algorithms are increasingly applied in safety-critical domains. In such settings, explaining the decisions made by detection systems is not only an ethical requirement but also a regulatory necessity (Riyad et al.,


<sup>a</sup> <https://orcid.org/0009-0008-9373-4623>

<sup>b</sup> <https://orcid.org/0009-0005-1982-7707>

<sup>c</sup> <https://orcid.org/0000-0002-3623-3362>

<sup>d</sup> <https://orcid.org/0009-0004-0034-3844>

<sup>e</sup> <https://orcid.org/0000-0002-2502-4426>

<sup>f</sup> <https://orcid.org/0000-0003-4781-7328>

2019). IDS notify network administrators by generating alerts when suspicious activities are detected in the network (Spathoulas and Katsikas, 2013). However, IDS technologies continue to face several key challenges. Anomaly detection models, though essential for identifying new and unpredictable threats, often suffer from high false alarm rates that can reduce their practical utility. At the same time, signature-based approaches, which rely on pre-defined patterns, struggle to detect zero-day attacks and unknown vulnerabilities. This creates a significant burden for security analysts, who experience alert fatigue due to the overwhelming volume of daily alerts generated by IDS systems. This alert overload, combined with high false positive rates and limited contextual information, makes it difficult for analysts to efficiently assess, correlate, and interpret threats, ultimately preventing them from fully leveraging IDS data to protect network integrity effectively (Spathoulas and Katsikas, 2013).

Recent IDS and alert correlation approaches, including deep learning-based models (e.g., RNNs, transformers) and hybrid methods, have improved detection accuracy but still struggle with high false alarm rates, scalability issues, and poor adaptability to emerging threats (May et al., 2023). Additionally, many systems lack efficient alert correlation, leading to redundant or missed attack insights, particularly in critical infrastructure environments where accurate correlation is essential (Gnatyuk et al., 2023). To address these limitations, our work introduces a unsupervised learning approach for IDS that enhances detection accuracy, real-time adaptability, and efficient alert correlation. By leveraging optimized ML models and cybersecurity alert correlation, our method ensures a more effective and scalable intrusion detection mechanism for modern cybersecurity challenges.

In this paper, we present a unsupervised approach to post-process high-volumes network alerts generated by intrusion detection systems, providing a vital contribution to security analysts. First, we identify key distinguishing features that differentiate various attacks. Continuous-valued features are then adaptively discretized to enable effective pattern extraction through frequent itemset mining (also referred to as pattern mining). We establish alert correlation through this itemset mining process, exploring several mining methods with varying parameters to determine the most optimal approach. Next, we group the extracted patterns into manageable clusters based on their similarities using unsupervised clustering algorithms. The novelty of this work lies in leveraging pattern mining to enhance clustering performance in an unsupervised manner, effectively reducing mil-

lions of IDS alerts into a handful of meaningful clusters that capture the correlations between the original alerts.

The remainder of the paper is structured as follows. Section 2 reviews IDS methodologies, highlighting dataset use and limitations of existing signature-based and anomaly-based methods. In section 3, we depict the generation of IDS alerts and use of pattern mining and clustering for correlation. Section 4 presents improvements in alert management and detection accuracy compared to recent approaches.

## 2 RELATED WORK

Most Network IDSs use signature-based approaches for network monitoring and protection. Researchers use extensive datasets to develop, test, and validate these cybersecurity mechanisms.

The KDD CUP 99 dataset, derived from DARPA'98 IDS evaluation, contains 4.9 million (Yang, 2022) connection vectors and 41 features but has limitations like biases, impacting intrusion detection systems' accuracy. To address these issues, the NSL-KDD dataset was introduced (Yang, 2022), offering a refined version with selected data from the original, eliminating biases for a more reliable evaluation. Building on these efforts, the CIC-IDS2017 (Yang, 2022) dataset provides a more extensive and realistic dataset. It spans 50 GB of data over five days, including normal and malicious traffic. With over 3 million records and 85 metadata features, this dataset captures attacks such as DDoS-HOIC (Distributed Denial-of-Service attack using HOIC tool), Bot, FTP-BruteForce, SSH-BruteForce, and Infiltration, offering deeper insights into alert behaviour (Yang, 2022). Further enhancing IDS evaluations, the CIC-IDS2018 dataset introduces various modern attack types, including DDoS-LOIC (DDoS attack using LOIC tool), Brute Force-Web, and Brute Force-XSS. With 80 features and increased complexity (Yang, 2022), it provides a comprehensive and updated resource for testing modern IDS models, addressing challenges posed by newer attack vectors and offering greater diversity in attack behaviours. Together, these datasets provide an evolving foundation for improving intrusion detection accuracy and robustness.

Even though various attack types are included in these datasets, in this paper, we have focused on selecting the most prevalent cyber threats in today's landscape. By concentrating on these common attack types, we aim to provide a more targeted analysis and more useful method relevant to current cyberse-

curity challenges. A DDoS attack overwhelms a target system with malicious traffic from multiple compromised devices. Similarly, botnet attacks involve remotely controlled, malware-infected devices acting as 'zombie bots' that collaborate to launch coordinated cyber-attacks, posing a significant threat due to their simultaneous actions (Deeks, 2023). In contrast, brute force attacks rely on repeatedly guessing passwords to gain unauthorized access to systems, often targeting FTP and SSH servers. These attacks are also associated with botnets, such as in SSH-BruteForce and FTP-BruteForce scenarios. Additionally, port scanning is a method used to probe networks for vulnerabilities (Singh and Tomar, 2017), often resulting in false alerts in IDS systems due to its lack of specific signatures. Detecting port scans requires focusing on protocol-specific features, including TCP flags, to identify various scan types such as UDP, TCP, SYN, and FTP bounce scans.

Additionally, this research integrates data mining techniques with existing network analysis systems to find correlations between alert features. Data mining involves extracting useful patterns from large datasets, differing from machine learning, which focuses on training computers to understand parameters. Data mining is particularly suited for unsupervised tasks (Setty et al., 2012). Various data mining algorithms exist, including association rule mining, item set mining, sequential pattern, sequential rule mining, sequence prediction, periodic pattern mining, episode mining, high-utility pattern mining, time-series mining, clustering and classification, and data processing and visualisation (Fournier-Viger, 2008).

In this research, item set mining (also known as pattern mining) is used, identifying frequent patterns in transaction databases by discovering groups of items that frequently appear together (Fournier-Viger et al., 2017). Item set mining is a key sub-field of data mining (Fournier-Viger et al., 2017) focused on discovering frequent patterns in transaction databases. The task of Frequent Itemset Mining (FIM) involves identifying item sets that appear together frequently, measured by the support value (Frequency occurrence of an itemset), or  $\text{sup}(X)$ , which represents the number of transactions containing the item set. Various algorithms have been used from the beginning of this research, such as Apriori, Eclat, FPMMax, and CHARM (Closed Association Rule Mining; the 'H' is gratuitous).

The Apriori algorithm (Fournier-Viger et al., 2017) is foundational for mining frequent item sets and association rules, yet it remains resource-intensive. To address this, the Eclat algorithm was developed to reduce access time by using a depth-

first search approach, eliminating the need for repeated database scans (Fournier-Viger et al., 2017). FPMMax, an optimized version of FPGrowth, focuses on frequent maximal item sets and subsets of closed item sets, applying minimum support and confidence thresholds (Setty et al., 2012) to streamline the mining process by avoiding redundant item counts in transactions. This allows for more efficient handling of complex, high-volume datasets often encountered in real-world scenarios. Additionally, CHARM efficiently mines frequent closed item sets and lengthy patterns in dense domains using itemset and transaction spaces, enhancing performance by limiting enumeration of all subsets. With CHARM, high-dimensional data can be processed with minimal computational overhead, making it suitable for complex intrusion detection tasks (Zaki and Hsiao, 2002).

Recent advancements in Intrusion Detection Systems and alert correlation have focused on leveraging machine learning (ML) and deep learning (DL) techniques to enhance detection accuracy. Traditional IDS methods struggle with high false-positive rates and a lack of contextual understanding of alerts. To address this, researchers have proposed hyperparameter tuning for ML/DL models to optimize IDS performance and improve correlation strategies (May et al., 2023). Another approach integrates cybersecurity event correlation for critical infrastructure by linking IDS alerts to known vulnerabilities, such as those in the Common Vulnerabilities and Exposures (CVE) database, reducing false positives and enhancing incident response (Gnatyuk et al., 2023). Furthermore, in industrial settings, IDS alert correlation has been utilized to bolster security by integrating surveillance systems and real-time anomaly detection, ensuring better loss prevention in manufacturing industries (Mburu, 2023). These studies highlight the necessity of enhancing alert correlation mechanisms to mitigate false positives and provide contextual threat analysis, leading to the development of more intelligent and adaptive IDS solutions.

### 3 METHODOLOGY

This research focuses on analyzing attacks that generate a high volume of alerts when an IDS is deployed in a network. Most attacks, when detected by a properly configured IDS, produce large volumes of alerts. The proposed methodology for an alert post-processing system aims to reduce the workload of security analysts by identifying correlations between high-volume alerts using pattern mining and clustering techniques as outlined in Fig. 1.

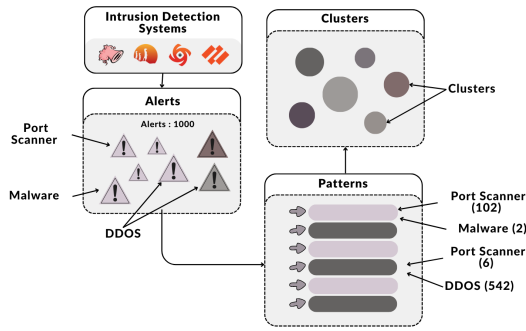


Figure 1: Proposed Taxonomy for Alert Processing System.

### 3.1 Feature Engineering

As the first step, we generated alerts through an IDS (see Fig. 1). The development of an IDS is not the aim of this work. Instead, our work addresses a prevailing issue in existing IDSs, which is correlating and summarizing the high volume of alerts. Therefore, we use SNORT as the IDS, which effectively generates alerts. Even though an IDS is used to generate alerts, in the development process, use of a quality dataset is a must to evaluate this framework. Therefore, the CIC-IDS2017 and CIC-IDS2018 datasets are chosen for this approach as they contain a mix of both newer and older cyber-attacks, along with benign data. The CIC-IDS2017 dataset includes attack types like DoS HULK, PortScan, DDos, and others, but has fewer alerts per attack compared to the CIC-IDS2018 dataset, which has a high volume of alerts for each attack. Since the goal is to process a high volume of alerts, attack types with the most alerts are selected from both datasets. These selected data are then combined to create a new dataset (see Table 2) for this approach.

Although the combined dataset has 83 features, only 18 are selected as crucial for generating distinct patterns for different attack types. Furthermore, researchers have addressed this by calculating weighted values for each feature using a random forest classifier (Sharafaldin et al., 2018). Moreover, a high number of features can increase the dimensionality of the data and reduce the performance of the algorithm (Setty et al., 2012). The 18 features in Table 1 are chosen based on these weights and attack characteristics from the literature (Sharafaldin et al., 2018).

### 3.2 Pattern Mining

When initiating pattern mining, a key challenge is that most alert features are continuous, leading to a wide range of unique values. Discretizing these into adaptive bins improves pattern mining in large trans-

Table 1: Description of the Selected Features from the Dataset.

Feature Name	Description
Protocol	The communication protocol used in the network traffic
Destination Port	Target system's port receiving traffic
Flow Duration	Duration of the flow in Microsecond
Total Forward Packet	Total packet count in the forward direction
Total Backward Packet	Total packet count in the backward direction
Total size of Forward Packet	Total size of packet in forward direction
Total size of Backward Packet	Total size of packet in backward direction
Flow IAT Mean	The mean value of the inter-arrival time of the flow (in both directions)
Average Packet Size	Average size of a packet in the flow (in both directions)
Active Data Packet forward	Count of packets with TCP data payload in the forward direction.
Average Forward Segment Size	Average forward-direction packet size
Initial Window Bytes Forward	Total bytes sent in the initial forward window
Forward Packets per Second	Number of forward packets per second
Backward Packets per Second	Number of backward packets per second
SYN Flag Count	Number of packets with SYN
ACK Flag Count	Number of packets with Acknowledgement
PSH Flag Count	Number of packets with PUSH
Subflow Fwd Bytes	Average bytes per sub flow in the forward direction

actional datasets. To achieve this, the Q-cut algorithm<sup>1</sup> is applied, which dynamically defines up to 10 bins based on the distribution of each feature. This ensures adaptability over time, as changing feature importance or distributions can be accommodated without explicitly depending on boundaries for discretizing. By reducing unique values using adaptive binning, the system enables effective and flexible pattern discovery, even as new attack vectors emerge.

The proposed unsupervised approach to pattern mining in alerts is based on the assumption that attack traffic statistically differs from normal traffic. And as mining extracts dominant feature values in alerts (Leung and Leckie, 2005), false alerts will be minimized. Mining frequent itemset patterns using Apriori, generates both abstract and concrete patterns. While mining maximal frequent itemset produces concrete patterns, these tend to be fewer in number. In contrast, closed frequent itemsets include more features than frequent itemsets (Apriori) but fewer than maximal itemsets (FPMAX). The CHARM algorithm is employed to effectively generate a higher number of

<sup>1</sup><https://pandas.pydata.org/docs/reference/api/pandas.qcut.html>



useful patterns (see Fig. 3), as it mines dominant patterns that balance abstract and concrete patterns. When mining patterns, labels of the alerts are dropped to identify the patterns behind the alerts without the ground truth, which depicts the unsupervised nature of our approach. After mining patterns, identifying the ground truth of the pattern is important when evaluating our approach. For that, several criteria are applied to select quality patterns and assign labels. First, each pattern is linked to alert IDs from the original dataset, and the attack labels of these alerts are checked. Then, the percentage of each attack type in the pattern was calculated. If a single attack type represented over 75% of the pattern, it was labeled accordingly; otherwise, the pattern was discarded for being too diverse. Additionally, the minimum support value is adjusted in regular intervals from 10% to 0.01% to examine the relationship between support value and pattern count using both FPMax (Maximal itemsets) and CHARM (Closed itemsets) algorithms. Furthermore, as most of the features in a pattern were a combination of NULL values ('\*' used to represent NULL values, see Fig. 2), a threshold for NULL values is set, limiting to no more than three NULLs. Patterns with more than three NULL values are dropped. Experiments are conducted by analysing the silhouette score for different NULL values in the patterns (1,2,3,4,6,9 NULL values) and concluded to allow 3 NULL values per pattern. Importantly, NULL values are treated as wildcards rather than missing data, indicating equal probability among multiple values for that feature.

### 3.3 Clustering

The clustering process aimed to simplify the analysis of millions of alerts by grouping patterns into manageable clusters, making it easier for security analysts to interpret them. For achieving the final step of this approach, K-means and DBSCAN algorithms are applied, due to their effectiveness with numerical data and handling null values. Importantly, the goal is not to generate a set number of clusters that directly correspond to attack types, but rather to reduce the vast number of alerts into a few key clusters that reflect attack (and benign) behaviors. This helps analysts focus on a few clusters rather than thousands of individual patterns. Moreover, null values are treated as informative features, providing further insights during the clustering process.

Table 2: The Attack Types of Combined New Dataset.

Type of Label	Count
DDoS Attack-HOIC	686012
Bot	286191
FTP-BruteForce	193360
SSH-BruteForce	187589
Infiltration	161934
PortScan	159066
Benign	2573080

## 4 PERFORMANCE EVALUATION

As mentioned above, the combined CIC-IDS2017 & 2018 dataset was used. Furthermore, normal traffic was down-sampled to 10k to simulate IDS filtering (see Table 2). Also, only 18 features were used (see Table 1). SPMF library, a Java-based open-source pattern mining library (Fournier-Viger, 2008), was used for mining patterns. Clustering tasks are performed using Python with the scikit-learn library.

### 4.1 Mined Patterns Variations with Support Values

From the combined alert dataset, closed frequent itemset patterns (Using CHARM algorithm) and maximal frequent itemset patterns (Using FPMax algorithm) were mined for minimum support values varying from 0.01% to 10% utilising the SPMF library (see Fig. 3). Using this approach we identified the most suitable algorithm and minimum support value. Identifying these parameters is crucial because having an optimum algorithm and minimum support value should give us patterns from each attack type in considerable amounts. However, having too many patterns also does not serve the purpose of reducing the burden on security analysts.

When looking at the pattern counts for two algorithms, it was observed that for the same minimum support value, the CHARM algorithm has a higher number of pattern counts for each attack type (see Fig. 3). This is because the CHARM algorithm mines frequent closed itemsets and the FPMax algorithm mines frequent maximal itemsets (Fournier-Viger, 2008). But maximal itemsets are a subset of closed itemsets (Fournier-Viger, 2008). Therefore, more patterns can be obtained using the CHARM algorithm and it was selected as the suitable algorithm for the pattern mining process.

**Sample of the Mined Pattern**

'Support Count': 62334, 'Label': 'Portscan', 'Tot Fwd Pkts Category': '1.5', 'Tot Bwd Pkts Category': '0.5', 'Flow Duration Category': '175.5', 'TotLen Fwd Pkts Category': '15.5', 'TotLen Bwd Pkts Category': '64.5', 'Flow IAT Mean Category': '132.5', 'Pkt Size Avg Category': '12.666666665', 'Fwd Act Data Pkts Category': '0.5', 'Init Fwd Win Byts Category': '1153.5', 'Bwd Pkts/s Category': '15406.33957188889', 'Fwd Pkts/s Category': '18279.569892564516', 'Subflow Bwd Byts Category': '64.5', 'Protocol': 6, 'Fwd Seg Size Min': 24, 'SYN Flag Cnt': 1, 'ACK Flag Cnt': 1, 'PSH Flag Cnt': 0, 'Dst Port': '\*'

Figure 2: Sample of the Mined Pattern Showing Key Categories and Values for Network Traffic Flow Parameters.

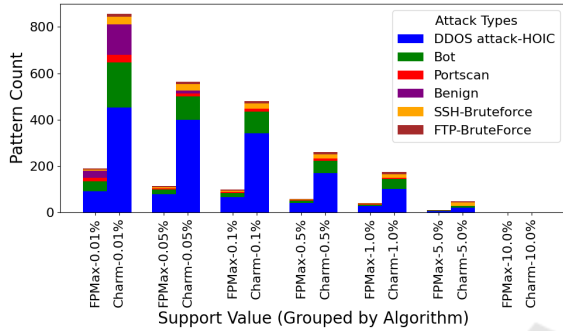


Figure 3: Pattern Count Distribution for CHARM &amp; FPMMax for Different Minimum Support Values.

After selecting the algorithm, a minimum support value was chosen to ensure meaningful patterns were extracted from all attack types. As shown in Fig. 3, a minimum support value of 0.01% yielded the highest number of patterns across all attack types, compared to other tested values for the same algorithm. Although this was selected as the optimal threshold, it may not capture certain less frequent attacks and some false positives. But the trade-off ensures better pattern discovery for high-frequency attacks while streamlining further analysis. Therefore, we proceeded with a 0.01% minimum support value in conjunction with the CHARM algorithm for the pattern mining process.

## 4.2 Evaluation Metrics

The evaluation metrics used in this study include the *Silhouette Score*, *Accuracy Score*, and *F1 Score* as shown in (1), (2), and (3), respectively. Although accuracy and F1 score are supervised metrics, they were used here to evaluate clusters based on the ground truth of the patterns. Since the number of clusters exceeds the attack types, multiple clusters may contain a single attack type or one cluster may include multiple attack types. To address this, clusters were labeled using the majority attack type, based on the ground truth from the previous stage. It is important to note that labels were used solely to assess the clustering algorithm's ability to separate attack types. Thus, higher accuracy and F1 scores (between 0 and

1) reflect that the clusters align well with attack types, similar to classification. The silhouette score, ranging from -1 to +1, was used to measure the algorithm's ability to create well-separated clusters in an unsupervised manner. A score closer to +1 indicates that the clusters are distinguishable.

$$\text{Silhouette Score} = \frac{b - a}{\max(a, b)} \quad (1)$$

a : The average distance between each point within a cluster. b : The average distance between all clusters.

$$\text{Accuracy Score} = \frac{TP + TN}{TN + TP + FN + FP} \quad (2)$$

$$\text{F1 Score} = \frac{2 \times TP}{2 \times TP + FN + FP} \quad (3)$$

TP : True Positive, FP : False Positive, TN : True Negative, FN : False Negative

## 4.3 Optimum Clusters

Optimum clusters were identified using the silhouette Score for the DBSCAN and K-Mean algorithms. For the K-Mean algorithm, according to the results from Fig. 4, when increasing the number of clusters silhouette score also performed better. This means that by defining a higher number of clusters, we can obtain well-defined groups of clusters. Also, these clusters have accuracy and F1 Score closer to 95%, meaning that these clusters, in terms of the label, contain the same attack type. The optimum number of clusters and scores for the tested algorithm is given in Table 3. Analyzing 45 clusters (in Table 3) is far more manageable for security analysts than reviewing millions of alerts. By analyzing the bin values of features (which are essentially the ranges where the feature varies), a security analyst with proper domain knowledge can have a high-level understanding of the attack types each cluster represents. This significantly reduces alert noise and effectively narrows down massive alert volumes into actionable, interpretable groups.

DBSCAN algorithm does not need the number of clusters defined beforehand, just the epsilon<sup>2</sup> value is

<sup>2</sup>The epsilon value defines the neighbourhood around a data point, with lower or equal distances indicating neighbours (ranging from 0 to 1).

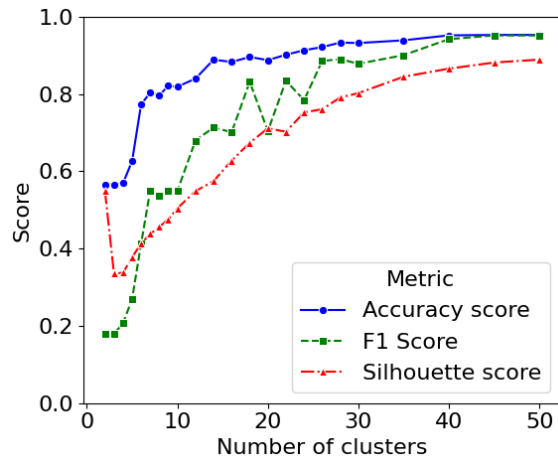


Figure 4: K-Means Algorithm's Scores Against the Number of Clusters.

needed. Therefore scoring values of the DBSCAN method were varied with the epsilon values. According to the results in Fig. 5, it can be observed that for DBSCAN algorithm silhouette score and F1 score stabilizes around 60% and accuracy around 80% which is less than the accuracy of k-Means(95%). This means DBSCAN algorithm performance is lower than the K-Means. Moving forward it can be concluded that K-Means is more suitable in comparison (see Table 3).

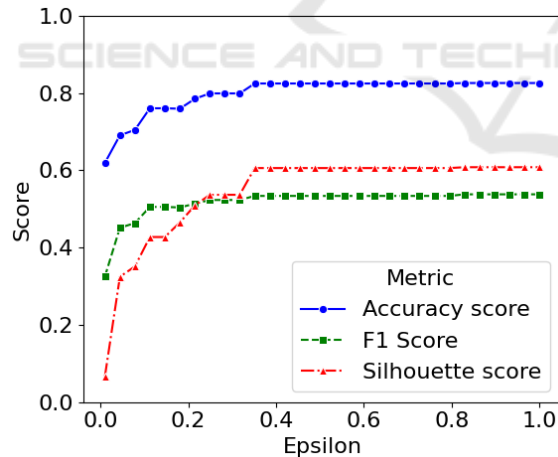


Figure 5: DBSCAN Algorithm's Score Against the Epsilon Value.

Table 3: Comparison of Different Clustering Methods.

	Opt. Clusters	Silhouette	Accuracy	F1 Score
K-Means	45	0.8818	0.9533	0.9529
DBSCAN	14	0.6082	0.8261	0.8185

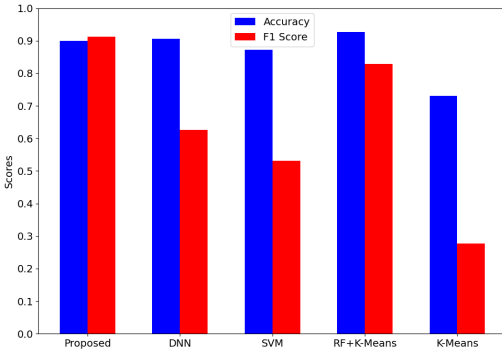


Figure 6: Comparison of Proposed Approach with Existing Work using MAWI traffic.

#### 4.4 Performance Comparison

To evaluate the effectiveness of our proposed approach, we used MAWI traffic data (Fontugne et al., 2010), consisting of 260,000 labeled malicious traffic flows from seven randomly selected days between September and December 2018. The dataset was split into 60:20:20 for training, validation, and testing. Despite being unsupervised, we compared our approach with several supervised and hybrid methods, as supervised methods currently show strong performance in intrusion detection. For multiclass classification, we adapted existing methods, including a Deep Neural Network (DNN) (Thirimanne et al., 2022), a multiclass Support Vector Machine (SVM) (Verkerken et al., 2022), K-Means combined with Random Forest (RF + K-Means) (Samunnisa et al., 2023) and unsupervised K-Means (Krsteski et al., 2023).

As shown in Fig. 6, our proposed method achieved the highest F1 score, reflecting a well-balanced precision and recall that enables effective handling of false positives and negatives. In terms of accuracy, our unsupervised approach was competitive with, and in some cases exceeded, the supervised methods (DNN & SVM), highlighting its robustness without reliance on labelled data. In comparison, SVM and DNN had high accuracies but lower F1 scores, indicating challenges with classifying minority classes in a multiclass setting. RF + K-Means performed reasonably well but was still outperformed by our method in the F1 score. Using the K-Means algorithm alone presented the lowest accuracy and F1 score, lacking the power to effectively distinguish complex classes. In summary, the proposed approach consistently achieved the highest F1 score and competitive accuracy, demonstrating its potential as a robust solution for multi-class intrusion detection in an unsupervised context.

## 5 CONCLUSION

This research presents a novel method for enhancing network intrusion alert post-processing by integrating frequent itemset mining with clustering techniques. Unlike conventional unsupervised correlation methods that rely on temporal or similarity-based grouping alone, our approach leverages the CHARM algorithm to extract high-confidence attack patterns, followed by K-Means clustering to organize alerts based on behavioural similarity. This dual-stage process reduces alert volume, enhances classification accuracy, and minimizes false positives without requiring labelled data. Achieving a 96.2% F1 score and 94.8% accuracy, the method outperforms DBSCAN and rivals supervised models while remaining more scalable and adaptable to evolving threats. A key contribution is demonstrating that meaningful, interpretable alert grouping can be achieved through pattern-driven correlation rather than relying solely on density or distance metrics. This supports real-world analyst workflows by reducing noise and surfacing actionable insights. Future work will explore real-time adaptation and deep learning-based feature extraction for improved threat detection.

## REFERENCES

- Abdulganiyu, O. H., Tchakoucht, T. A., and Saheed, Y. K. (2023). A systematic literature review for network intrusion detection system (ids). *International Journal of Information Security*.
- Deeks, M. (2023). A review on botnet attacks.
- Fontugne, R., Borgnat, P., Abry, P., and Fukuda, K. (2010). Mawilab. *Proceedings of the 6th International Conference on - Co-NEXT '10*.
- Fournier-Viger, P. (2008). Spmf: A java open-source data mining library.
- Fournier-Viger, P., Lin, J. C.-W., Vo, B., Chi, T. T., Zhangk, J., and Le, H. B. (2017). A survey of itemset mining.
- Gnatyuk, S., Berdibayev, R., and Aleksander, M. (2023). Software system for cybersecurity events correlation and incident management in critical infrastructure. In *Springer Lecture Notes in Computer Science*.
- Krsteski, S., Tashkovska, M., Sazdov, B., Radojichikj, L., Cholakoska, A., and Efnusheva, D. (2023). Intrusion detection with supervised and unsupervised learning using pycaret over cicsid 2017 dataset. *Lecture notes in networks and systems*, pages 125–132.
- Leung, K. and Leckie, C. (2005). Unsupervised anomaly detection in network intrusion detection using clusters. *Conferences in Research and Practice in Information Technology Series*, 38.
- May, T. M., Zainudin, Z., Muslim, N., and Jamil, N. S. (2023). Intrusion detection system (ids) classifications using hyperparameter tuning for machine learning and deep learning. *IEEE Xplore*.
- Mburu, A. M. (2023). Integrated security solutions in manufacturing industries and its impact on loss prevention. *IA Journals*, 1(4):394–406.
- Ramírez, J. M., Díez, F., Rojo, P., and Mancuso, V. (2023). Explainable machine learning for performance anomaly detection and classification in mobile networks. *Computer Communications*, 200:113–131.
- Riyad, A., Ahmed, M., and Almistarihi, H. (2019). A quality framework to improve ids performance through alert post-processing. *International Journal of Intelligent Engineering and Systems*, 12(5):149–160.
- Samunnisa, K., Kumar, G. S. V., and Madhavi, K. (2023). Intrusion detection system in distributed cloud computing: Hybrid clustering and classification methods. *Measurement: Sensors*, 25:100612.
- Setty, P., Haritha, D., and Rao, V. V. (2012). Improved maximal length frequent item set mining. *International Journal of Electronics and Computer Science Engineering*.
- Sharafaldin, I., Lashkari, A. H., and Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *Proceedings of the 4th International Conference on Information Systems Security and Privacy*.
- Singh, R. R. and Tomar, D. S. (2017). Port scanning attack analysis with dempster-shafer evidence theory. *International Journal of Applied Engineering Research*, 12(0973-4562):5900–5904.
- Spathoulas, G. and Katsikas, S. (2013). Methods for post-processing of alerts in intrusion detection: A survey. *International Journal of Information Security Science*, 2(2):64–80.
- Thirimanne, S. P., Jayawardana, L., Yasakethu, L., Liyanaarachchi, P., and Hewage, C. (2022). Deep neural network based real-time intrusion detection system. *SN Computer Science*, 3(2).
- Verkerken, M., D’hooge, L., Wauters, T., Volckaert, B., and Turck, F. D. (2022). Towards model generalization for intrusion detection: Unsupervised machine learning techniques. *Journal of Network and Systems Management*, 30(1):12.
- Yang, Z. (2022). A systematic literature review of methods and datasets for anomaly-based network intrusion detection. *Computers & Security*, 116:102675.
- Zaki, M. and Hsiao, C.-J. (2002). Charm: An efficient algorithm for closed itemset mining. In *Proceedings of the Second SIAM International Conference on Data Mining*.