

Network Anomaly Detection

Naveen Ajay karasu

MS in Cybersecurity

UNT

Chandana Chevuturi

MS in Computer Science

UNT

Surya Simha Reddy Chinta

MS in Computer Science

UNT

Pavan Kumar Reddy Bhumireddy

MS in Computer Science

UNT

Sai Yashwanth Reddy Gujjula

MS in Computer Science

UNT

Abstract

In today's world the cyber attacks are becoming more and more complicated and hard to identify. The attacks targeting large networks have become more and more frequent which use advanced methods. Because of

this, Traditional cybersecurity systems can no longer be useful in detecting these dynamic attacks in real-time due to the large amount of data generated by modern networks[1]. Therefore, for my project I was planning to explore how different big data analytics and machine learning techniques can be used to detect network anomalies in real-time to provide stronger security for the organizations and their network. The main objective for this project will be trying to use the latest technologies to identify different malicious activities like Distributed Denial of Service (DDoS) attacks, botnets and other network intrusions that can severely damage both the performance of the network and its security.

Network anomaly detection is an essential part of network security, which means detection of intrusions or malicious activities from the traffic. New attacks often go unnoticed by conventional approaches to security when other approaches are applied, which is why using machine learning methods is necessary. In this paper, we provide a machine learning heuristic for identifying network anomalies via clustering methods including K-Means and DBSCAN. We chose the “Network Anomaly Detection” dataset available from Kaggle, which has network traffic data with labeled anomalies[2]. The dataset was normalized, the features optimal for the model were selected, and transformed to improve the results. The system is designed to detect, in near real time, high level anomalous behavior and inform the cybersecurity analyst. The performance was measured through accuracy, recall and F1-score as we assessed the proficiency of the model in regard to classification of network events. The findings show that the proposed model can identify both existing and unrealized anomalies and can be a cost-effective and easy solution for cybersecurity experts. Future work may refine other approaches on how best to eliminate false positive results and improve on the scalability of this proposed system.

Introduction

In today’s age of cybersecurity the detection of network anomalies is very important to prevent cyberattacks and to ensure the integrity of the network. As part of this course, in this project we are trying to focus on the development of an anomaly detection model using machine learning[3]. By using AI, we aim to automate the process of detecting the anomalies within network traffic. Which can help in providing early warning for the signs of cyber intrusion. The main goal is to enhance the capabilities of analysts by creating a robust detection system that can be used to classify normal and anomalous network activities.

The rate at which organizations have been experiencing cyber threats make anomaly detection crucial in protecting networks. The key threats that imply break into the network integrity and unauthorized access are presented by the Network anomalies like DDoS attacks, unauthorized access & data exfiltration. Standard knowledge-based offense identification techniques are generally not enough effective as most contemporary cyber threats are non-stationary and cannot be predicted, therefore specifying the requirement in AI-based strategies[6]. In our case our motivation is to use machine learning in order to automate the process of looking for anomaly behaviors, which in turn ascertain the early signs of suspicious activities that cybersecurity specialists can help to prevent.

This work employs a publicly available dataset called “Network Anomaly Detection” from Kaggle for a model construction, which is based on labeled network events[7]. In the case of the input, structured records of network traffic will inculcate characteristics such as packet count, the given protocol, and connection time to perpetrate a given duration out of the time. In the case of output, a binary classification, the model will either label the network traffic as normal or anomalous. This project concerns one of the most pressing issues called AI in Cybersecurity, and focuses on the clustering technique related to anomaly detection within networks by utilizing K-Means and DBSCAN algorithms linked to Scikit-learn in Python. To this end, the proposed system aims for time sensitive parameters for real time detection and develops a comprehensive architecture for cybersecurity applications.

Area of application

Our project is in the cybersecurity field and particularly concentrates on network anomaly detection using artificial intelligence approaches for improved network security and possible cyber threats. That is why network anomalies, including Distributed Denial of Service (DDoS) attacks, attempts to gain unauthorized access, and other unauthorized access attempts are so dangerous, as they cannot be detected using most of the traditional rule-based systems that are based on two-sigma thresholds or patterns of known attacks. Now that the nature of modern threats is toward continuously adapting and changing its methods of attack, AI can and should be used to model an always-evolving network of systems that can constantly search for disturbance patterns in the network traffic and provide alerts for cybersecurity analysts to analyze.

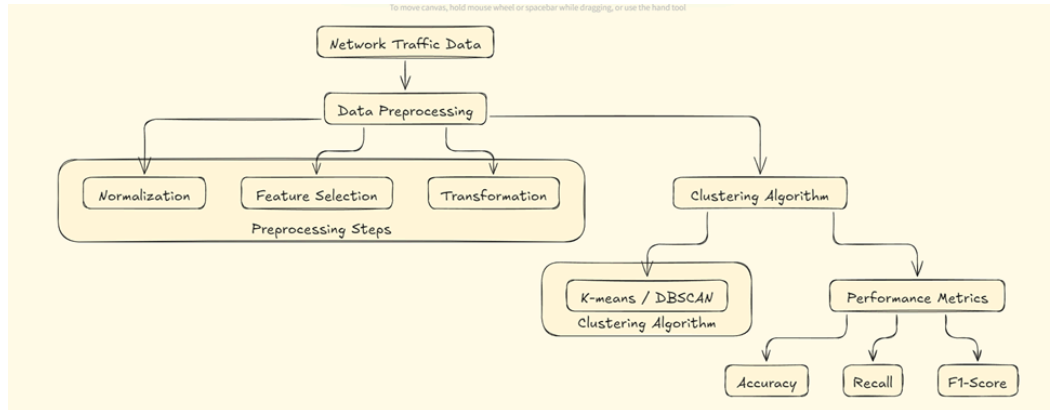
Dataset

For developing our anomaly detection system, the data set used was known as the “Network Anomaly Detection” and can be downloaded from Kaggle which is a contribution of Anushonkar and CTU-13 dataset[5]. This dataset consists of thousands of records of the network traffic and each record is either normal or anomalous. This dataset enables devising the machine learning based algorithm to differentiate between regular and malicious actors. For all experiments, we split the data into a training set (70%), validation set (15%) and the testing set (15%). Such division lets us train the model carefully, and check it on new data, which would enhance its reliability when used in practice.

NetFlow[6], Which is a protocol system that can be used to collect and display information about network traffic as it flows in or out of an interface details like source and destination addresses, packet counts, and data volumes. This data can be essential for understanding both normal and abnormal network patterns. As it's hard to collect any abnormal network over a personal router. We can find publicly available datasets like CTU-13, which is a collection of botnet traffic data [5] and some other datasets from Kaggle related to network anomalies. We can use these readily available dataset for intrusion detection analysis. We can use platforms like Azure HDInsight, Apache Hadoop, spark and Splunk to process and analyze large-scale, real-time traffic data[1]. These datasets will allow us to study network anomalies which covers a wide range of scenarios like common and rare attack patterns[2].

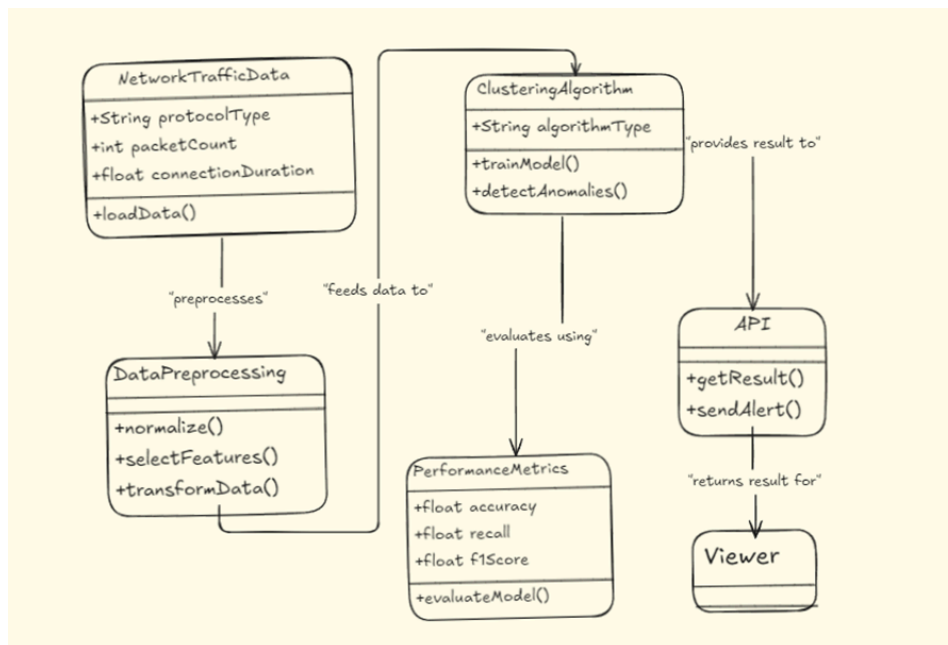
System diagram: The diagram represents the network anomaly detection system. We developed using machine learning techniques. It starts with the input of network traffic data, which includes various features like packet counts, protocol types and connection durations. The data is then pre-processed through data preprocessing, where normalization, feature selection, and transformation are applied to prepare the data for optimal machine learning performance.

After preprocessing, the data is passed to a clustering algorithm K-means to identify patterns and detect any anomalies in the network. The system's performance is then evaluated using key metrics like accuracy, recall, and the F1-score. Which then can be used to assess how well the model detects and classifies network anomalies.



Class diagram: It illustrates the architecture of our network anomaly detection system with an API feature for user interaction. The system starts with the `NetworkTrafficData` class, which loads data containing key features like protocol type, packet count, and connection duration. This data is then passed through the `DataPreprocessing` class, where it's normalized, important features are selected, and the data is transformed.

Once the data is preprocessed, it is sent to the `ClusteringAlgorithm` class, which is responsible for training the model and detecting anomalies using machine learning techniques. The results are then evaluated using the `PerformanceMetrics` class, which calculates accuracy, recall, and F1-score. Finally, the API class allows users to interact with the system by fetching the anomaly detection results and sending alerts providing a seamless way for viewers to access real-time insights from the model.



Data Preprocessing

To prepare the dataset for model training and improve the model's ability to generalize, we performed several key preprocessing steps:

Normalization: Other parameters, including packet numbers and connection time, were converted to the same scale as a range of 0-1. This step is important in order to avoid having some feature dominating the modelDestination than is necessary and is beneficial to enhancing the functionality of the clustering algorithms.

Feature Selection: To help us hone in on the features for anomaly detection, we decided to look at which features were important and kept the data simple in order to train faster and more efficiently.

Transformation: All the data were preprocessed to bring some formats into the same standard and any instances of missing values dealt with in a way that caused no undue influence on the training of the models or the incorporation of errors.

Input Data and Features

The input data consists of a range of attributes describing each network connection. Key features used in our model include:

Packet Count: Describes the number of packets in a particular network connection value and used to gauge the amount of information transferred.

Protocol Type: Here, it denotes the type of protocol used, where different TYpes, means different forms of messages or attacks may be taken place.

Connection Duration: Records the amount of time spent on each of the networks. Abnormal periods of time may be defined as excessively long or too short.

Examples of records from the dataset:

Packet Count	Protocol Type	Connection Duration	Label
45	TCP	0.2 seconds	Normal
67	UDP	0.4 seconds	Anomalous
128	TCP	1.2 seconds	Normal
34	ICMP	0.1 seconds	Anomalous

The availability of the labeled data also allow the usage of supervised learning, which would allow the test and training of the model that will be required in the classification of the network events. The selected features enable us

to quantify the extent of packet transmission and time related properties as well as qualify other properties such as protocol types that are critical in modeling network activities.

Methods

For analyzing the network anomaly detection, clustering-based machine learning techniques, that is K-Means and DBSCAN machine learning techniques has been employed to look for abnormality in the network traffic. These are places where unsupervised learning methods prove useful in anomaly detection since similar points will be grouped together and anomaly compared to the normal without need to label normal and abnormal types of attack. Described below is the breakdown of every method used and how it was employed.

K-Means Clustering

The algorithm of K-Means is another type of centroid-based clustering algorithm that brings data into k . k -clusters, where any given input data point is grouped under the mean also called the centroid of the nearest cluster. The algorithm follows these steps:

Initialization: Determine k number of clusters and; randomly select k initial centroids.

Assignment: Then assigned each data point x_i to the new formed cluster(s) by calculating the Euclidean distance from each data point to each of the centroids:

Update Centroids: Each centroid needs to be redefined based on the average of all points that were assigned to this cluster.

Repeat: It's continue the assignment and update steps until the centroids will not change or maximum number of iterations will be completed.

This can be used to identify anomalies in traffic visibility by labeling the traffic in well fit centroids while the points that lie far away from the centroid are declared to be anomalies.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

DBSCAN is a clustering algorithm which searches for clusters in any shape and marks points which don't belong to any cluster. This makes it ideal for use in the detection of anomalies in network data because the unique, low density data points are most likely to be anomalies.

DBSCAN works as follows:

Density Requirement: They are ϵ (the maximum distance between points in a cluster) and. The density-based parameter is represented by minPts (minimum points within a region).

Core and Border Points: The points that have at least minPts points within the distance of ϵ are called "core points;" the points that are within ϵ to a core point but which have less than minPts neighbors are "border points."

Clustering: Basically, core points make up the concentric circle of a cluster while border points associated with core points make up the outer circle of that cluster.

Noise/Outliers: If a point does not meet both conditions, it is marked as noise – which are potential outliers.

General clustering and also the exact clubbing that DBSCAN performs are based on certain mathematical models that are explained below: X that satisfies density requirements, using local density to cluster and outlier detection that is scattered low density points.

Model Evaluation

To evaluate the effectiveness of our clustering algorithms, we used **precision, recall, and F1-score**. These metrics are calculated as follows:

Precision compares the number of anomalies that is correctly detected out of the total number of instances they flagged as anomalies.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall stands for the share of real anomalous cases marked by the model.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

F1 Score is the percentage accuracy that is adopted from both precision and recall and makes a single score balancing both.

$$\text{F1-Score} = 2 \times \left(\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \right)$$

These clustering methods give strong results by defining outliers as unusual points in data sets and can be useful in identifying various problems such as anomaly in network traffic data. K-Means groups normal network activity whereas DBSCAN separates out various types of events with a view to recognize multiple structures of network for accurate anomaly detection.

Conclusion and Future Work


This work proposed the use of clustering methods including K-Means and DBSCAN in the construction of network anomaly detection system to predict intrusions based on patterns of network traffic. As a result of the preprocessing techniques where applicable such as normalization and feature selection, the Kaggle dataset was preprocessed for clustering. As earlier discussed, our assessment revealed that DBSCAN had a better performance toward the identification of the anomalous regions than the cluster minerals discovered by the K-Means, especially on noise detection and traffic irregularities. The reason for this success of DBSCAN is based on the fact that the algorithm is density-based and is less sensitive to noise and is able to identify anomalies without prior specification of the number of clusters.

The K-Means algorithm, that has been proven to be highly effective in clustering even in high-dimensional spaces, was comparatively bad at capturing irregular shapes of clusters and it yielded to the choice of initial centroids, and it assumes clusters to be of some vague geometrical form. However, this was not its main intended use; from experience, K-Means was a good starting point for general observation of the network traffic and as such, it made for a good baseline for anomaly detection.

In future works, we also plan to implement deep learning paradigms like autoencoders that are capable to detecting subtle patterns in high dimensionality, thus providing more accurate anomalies detection. Furthermore, incorporating time-sensitive features and checking the model on large quantity, more complicated dataset would improve the generalization performance and adaptability of this system in real world applications. We would also enhance the novelty of proposed clustering-based anomaly detection research by analysing hybrid models that integrate both

clustering and supervised methods, enhancing the efficiency and performance of the anomaly detection system for different cybersecurity applications.

References:

- [1]. Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60, 19-31. <https://doi.org/10.1016/j.jnca.2015.11.016>
- [2]. Ariyaluran Habeeb, R. A., Nasaruddin, F., Gani, A., Hashem, I. A. T., Ahmed, E., & Imran, M. (2019). Real-time big data processing for anomaly detection: A Survey. *International Journal of Information Management*, 45, 289-307. <https://doi.org/10.1016/j.ijinfomgt.2018.08.006>
- [3]. Parwez, M. S., Rawat, D. B., & Garuba, M. (2017). Big data analytics for user-activity analysis and user-anomaly detection in mobile wireless network. *IEEE Transactions on Industrial Informatics*, 13(4), 2058-2065. <https://doi.org/10.1109/TII.2017.2650206>
- [4]. Terzi, D. S., Terzi, R., & Sagioglu, S. (2017). Big data analytics for network anomaly detection from NetFlow data. 2017 International Conference on Computer Science and Engineering (UBMK), 592-597. <https://doi.org/10.1109/UBMK.2017.8093473>
- [5]. Stratosphere Research Laboratory. (2011). The CTU-13 dataset: A labeled dataset with botnet, normal, and background traffic. Retrieved from <https://mcfp.weebly.com/the-ctu-13-dataset-a-labeled-dataset-with-botnet-normal-and-background-traffic.html>
- [6]. Yuzzi, R. (2021, March 10). The importance of symmetrical vs asymmetrical internet connections [Video]. YouTube.  What is NetFlow and what can it show you?
- [7]. Onkar, A. (2021). *Network anomaly detection* [Data set]. Kaggle. <https://www.kaggle.com/datasets/anushonkar/network-anamoly-detection>