ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

# Network Traffic Monitoring Algorithm Based on Big Data Analysis

**Du Zhongxing**

*Dalian University of Technology, Dalian, 116024, China*

***Abstract:*** *Due to the ever-increasing variety of network threats and the need to store and query data in real time, conventional network traffic monitoring systems are becoming inadequate. As a result, the task of figuring out how to properly monitor massive amounts of network traffic has risen to the forefront of the field of network security management. In order to achieve this goal, we have proposed a new network monitoring system that uses big data technology and uses Netflow as the monitoring object. This system has four primary functions: it can utilize Filebeat to collect Netflow in real time; it transmits the data safely based on Logstash; it stores the data in ElasticSearch; and it analyzes and presents the data in real time using Kibana. The results of our experiments demonstrate that our system can provide millisecond-level replies to 100 million Netflows. It's able to give the foundation for network security management and fulfill the need for real-time monitoring of massive amounts of network traffic.*

***Keywords:*** *Big Data Analysis, Traffic Monitoring, Feature Extraction and model building*

## 1. Introduction

### *1.1 Overview of Big Data Analysis*

Big data analysis has become an essential tool for traffic monitoring, providing insight into the behavior of drivers and the performance of the traffic infrastructure. Big data analysis is used to identify patterns in traffic flow, detect potential problems, and develop solutions to improve traffic flow. Big data analysis is also used to design traffic monitoring algorithms that can be used to identify and analyze traffic patterns and detect potential problems. This report will discuss the design of a traffic monitoring algorithm based on big data analysis.

The report will begin with an overview of big data analysis and its importance in traffic monitoring. It will discuss the various techniques used to analyze big data and the advantages and challenges of using big data analysis for traffic monitoring. It will then discuss the process of designing a traffic monitoring algorithm based on big data analysis. This will include an overview of the algorithms and techniques used to analyze traffic data, as well as a discussion of the challenges and considerations involved in designing an effective traffic monitoring algorithm.

Finally, the report will discuss the implementation of the traffic monitoring algorithm. This will include a discussion of the various steps involved in the implementation process, as well as an overview of the technologies used to implement the algorithm. The report will conclude with a summary of the design of the traffic monitoring algorithm and its implementation.

This report will provide an overview of the design of a traffic monitoring algorithm based on big data analysis and discuss the implementation of the algorithm. It will provide insight into the challenges and considerations involved in designing and implementing an effective traffic monitoring algorithm and discuss the advantages and challenges of using big data analysis for traffic monitoring

### *1.2 Overview of Traffic Monitoring*

The Design of Traffic Monitoring Algorithm based on Big Data analysis is a comprehensive report on the development of an algorithm for traffic monitoring. This report provides an overview of the process of developing the algorithm, including the various datasets and tools used, as well as the overall design of the algorithm.

The purpose of traffic monitoring is to identify, analyze, and assess the traffic patterns in a given area. The data collected from traffic monitoring can be used to inform decisions related to traffic management,

ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

such as the deployment of traffic controllers or the installation of traffic signals. The development of a traffic monitoring algorithm is a complex task that requires an understanding of data analysis techniques, engineering principles, and urban planning.

This report is organized into several sections. The first section provides an overview of the traffic monitoring process and the datasets used. The second section discusses the design of the traffic monitoring algorithm and the tools used to implement it. The third section explains the testing and evaluation of the algorithm and the results obtained. Finally, the fourth section provides a summary of the findings and conclusions.

The development of a traffic monitoring algorithm is a complex and multi-faceted process. It requires an understanding of the underlying data and the tools available for implementing the algorithm. This report provides an overview of the process, from data acquisition to algorithm design, and includes an evaluation of the results. In addition, the report provides a summary of the findings and conclusions

## 2. Literature Review and Definition of Traffic Monitoring

In today's rapidly growing urban cities, traffic congestion has become one of the most pressing issues. The current traffic monitoring system is not able to effectively analyze traffic data and provide insights for better management of traffic. There is a need for a new traffic monitoring algorithm based on Big Data analysis that can provide better insights to address the traffic congestion issue. The proposed algorithm should be able to identify patterns in the traffic data, analyze the data in real time, and provide predictive analytics to enable better traffic management decisions. The algorithm should also be able to identify potential hazards and alert the traffic control systems. Furthermore, the algorithm should be able to integrate with existing traffic systems and be easily scalable with minimal training and maintenance costs.

The objective of this project is to design a traffic monitoring algorithm based on Big Data analysis that can be used to effectively monitor and manage traffic in urban cities. The proposed algorithm should be able to analyze traffic data in real-time, identify patterns, and provide predictive analytics for better decision-making. The algorithm should also be able to integrate with existing traffic systems and be easily scalable.

The problem definition of the traffic monitoring section of the report on The Design of Traffic Monitoring Algorithm based on Big Data Analysis is to design an algorithm to effectively monitor, analyze, and predict traffic patterns in real time. This algorithm should be able to utilize big data analysis to detect and alert traffic events, allowing for better decision-making and response to traffic incidents. The algorithm should be able to handle large amounts of data from multiple sources, such as vehicle-to-vehicle and vehicle-to-infrastructure communication (V2V, V2I), as well as from standard traffic monitoring devices, such as cameras, radar, and lidar sensors.

The algorithm should also be able to use machine learning and artificial intelligence techniques to accurately identify and recognize traffic patterns, such as lane changes, speed limits, and traffic sign violations, in order to provide accurate traffic predictions and alerts. The algorithm should also be able to integrate with existing traffic management systems and be able to adjust parameters in response to changes in the environment. Finally, the algorithm should be able to provide real-time visualization of traffic patterns, enabling better decision-making and improved traffic safety.

## 3. Methodology

### *3.1 Data Collection and Pre-processing*

The algorithm-data collection and pre-processing phase of designing a network traffic monitoring algorithm based on big data analysis is crucial to the effectiveness of the algorithm and the accuracy of the data analysis. This phase involves collecting, organizing, and pre-processing the data to be used in the analysis.

The first step in this process is collecting the network traffic data. This data can be collected from various sources, including network devices (such as routers, switches, and firewalls), network management systems, and other sources such as the Internet. The data collected should include the source and destination IP addresses, the port numbers, the packet size, and the time the packet was sent and received. This data must be collected in a manner that ensures accuracy and completeness. The second

ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

step is to organize the data. This involves sorting the data into different categories, such as source and destination IP addresses, port numbers, and packet sizes. This organization will help to ensure that the data is properly analyzed and used in the algorithm. The system for analyzing network data is composed primarily of five modules. The first module is the data acquisition module, which is responsible for collecting various types of network data, such as traffic data packages, equipment logs, SNMP information, equipment information, and server probe information. The second module is the data receiving module, which performs verification, cleaning, conversion, and reduplication operations on the collected data, and transforms it into the required data format for the system. The third module is the data storage module, which is responsible for the persistent storage of data. The fourth module is the data processing module, which mainly utilizes machine learning algorithms to predict and analyze the collected data. Finally, the fifth module is the data display module, which displays the results of the data analysis using various analysis algorithms. The display method used by the data display module is in line with the mainstream WEB front-end technology.as shown in Fig. 1.

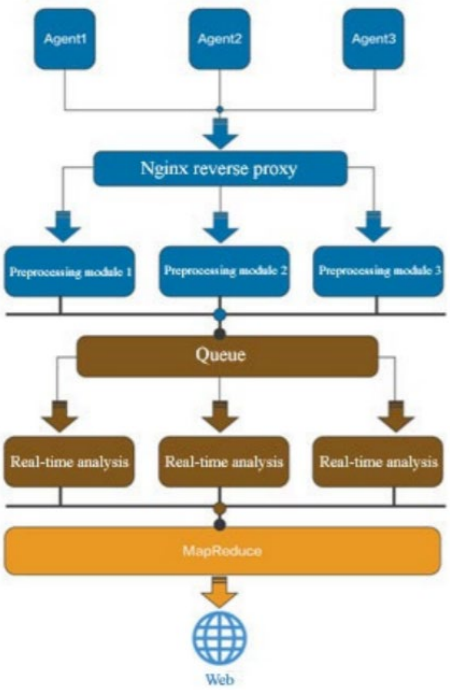


*Figure 1: System for Monitoring Network Traffic Using Big Data*

The final step is to pre-process the data. This involves cleaning the data to ensure that it is accurate and complete. It also involves normalizing the data to ensure that all of the data points are consistent. This includes removing any erroneous or incomplete data points, as well as ensuring that the data is in the same format.

Once the data has been collected, organized, and pre-processed, it is ready to be used in the design of the network traffic monitoring algorithm. The algorithm must be designed to analyze the data and identify any anomalies or patterns that may be present. By doing this, it can help to detect and prevent malicious activity, as well as identify potential bottlenecks in the network. It is important that the algorithm is designed to be efficient and accurate, so that the data analysis can be completed quickly and accurately.

### *3.2 Method for Feature Selection*

A preprocessing approach called feature selection (FS) should be used before data mining methods. Feature selection is a method for enhancing the effectiveness of data mining tools by removing unnecessary or unused information. By choosing just a portion of the original characteristics, feature selection techniques create a new set of attributes.

In order to improve network traffic analysis, feature selection is primarily used to minimize the dimensionality of the data collection. We discuss numerous preprocessing methods that scientists use prior to doing network traffic analysis. We have found certain methods that are often used for preprocessing network traffic data, such as principal component analysis, information entropy, rough set

ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

theory, and feature selection.

### *3.3 Feature Extraction*

Feature extraction is an important step in developing a network traffic monitoring algorithm based on big data analysis. The goal of feature extraction is to take a large, complex dataset and reduce it to a set of features that can be used to identify patterns in the data. The features that are chosen should be informative and provide the most useful information for understanding the data.

The first step in feature extraction is to identify the data that is relevant to the task. This can be done by looking at the data and determining which features are most likely to be useful in understanding the patterns in the data.

Once the relevant data is identified, the next step is to extract the features from the data. This can be done by using algorithms such as principal component analysis (PCA) or by manually selecting features. Once the features have been extracted, they can be used to train a machine learning algorithm. This can be done by using supervised or unsupervised learning algorithms. Supervised learning algorithms require labeled data, while unsupervised learning algorithms can be used to identify patterns in the data without labels.

Finally, the extracted features can be used to create a model of the network traffic. This model can then be used to detect anomalies or to identify trends in the data. This model can also be used to make predictions about future traffic.

By using feature extraction, a network traffic monitoring algorithm can be designed to more accurately detect patterns in the data and to make more accurate predictions. Feature extraction is an important step in designing a network traffic monitoring algorithm based on big data analysis.

### *3.4 Model Building*

Model building is the process of creating an algorithm that can accurately detect and classify a given type of network traffic. This process involves researching the properties of different types of network traffic, as well as identifying the most effective algorithms for detecting and classifying it.

The first step in model building is to identify the type of network traffic to be monitored. This includes defining the type of traffic to be monitored, such as web traffic, streaming video, or peer-to-peer file sharing. It also includes defining the traffic's characteristics, such as its protocol, size, and frequency [1].

Once the type of traffic is determined, the next step is to identify the most effective algorithms for detecting and classifying it. This involves researching algorithms that have been designed for similar types of traffic, as well as exploring new algorithms that could be developed to better detect and classify the traffic. This can include exploring machine learning techniques, such as artificial neural networks, to build an algorithm that can identify and classify network traffic.

The next step is to test the algorithm. This involves testing the algorithm on a simulated network to check its accuracy and effectiveness [2]. This testing should also include testing the algorithm on real-world data to ensure that it works as intended.

We modularize the system by abstracting it into four layers data collection, transport, storage, and analysis and display to increase the system's scalability and reliability. Figure 1 depicts the system architecture [3].Each component's functional design is outlined below: a. Nginx serves as a centralized data interface, receiving collected data via the HTTP protocol and converting it to JSON format. It distributes the data to the preprocessing module component group to balance the load. b. The preprocessing module formats and preprocesses the received source data, such as modifying field names, adding data types, and timestamps. It then forwards the processed data to the message queue. c. The message queue buffers the data to enable batch processing for real-time analysis module component group reading. d. The real-time analysis module retrieves the data from the cache, parses the fields, and stores the parsed results in the file system. e. The MapReduce framework is used for data storage and query. f. The web interface provides data visualization and human-computer interaction. It translates application requirements into complex query syntax to query and combine data.

System Framework The system framework is illustrated in Figure 2.

ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

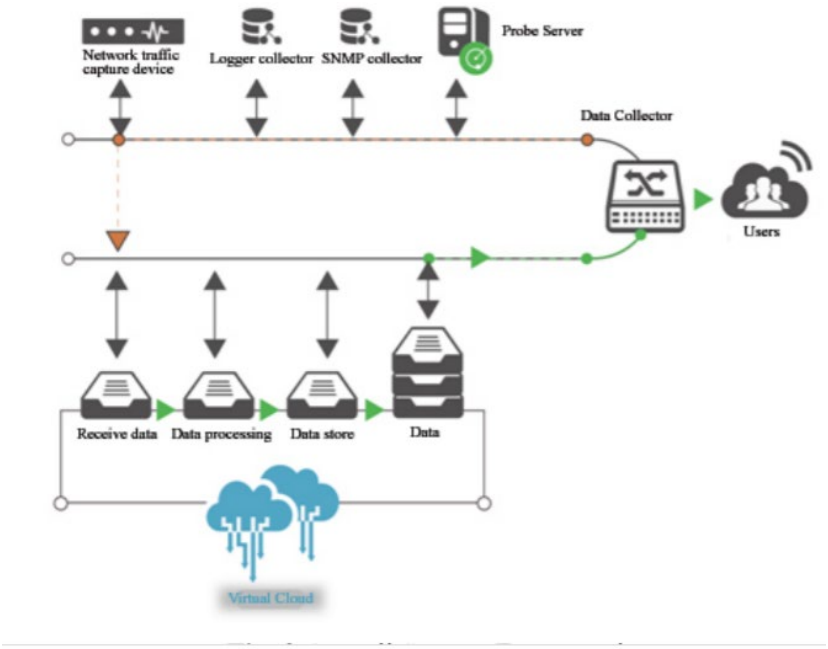


*Figure 2: Model of the system*

Data Acquisition Module The data acquisition module consists of hardware probes for network traffic, software probes, SNMP probes, and log collection services. These methods, including bypass mirror (Tap or Span), local monitoring (Local mode), and probes, can parse packets into more meaningful flow information.

Finally, the last step is to deploy the algorithm [4]. This involves deploying the algorithm on a live network and monitoring its performance. This will allow for further refinement of the algorithm as needed.

Once the algorithm is fully developed and tested, it can be used to detect and classify network traffic. This can be used to detect suspicious traffic, or to identify common patterns in network traffic. It can also be used to detect malware and other malicious activity [5]. This can help network administrators improve network security and ensure that the network is running smoothly.

### *3.5 Method of Clustering*

Clustering is the practice of dividing information into subsets defined by shared features. The process of clustering divides information into subsets based on shared characteristics. There are people who are highly similar in every group (a "cluster"), and others who are quite dissimilar in every other cluster. To analyze network traffic, clustering techniques are employed to put together disparate data sets. We found that academics use a variety of different clustering data mining strategies. Network traffic data mining hierarchical clustering was suggested by [6]. As a first step, they clustered the data streams representing network traffic using the K-means partitioning technique.

After locating the right cluster, further clusters are obtained through hierarchical approaches. In terms of traffic analysis, the authors claim that the bi-section K-means clustering result is superior to the pure hierarchical clustering technique [7].

Bai Using the Auto Class tool for data mining and K-means algorithms, [8] present a method for assessing the effect of wireless mesh network performance. The authors collected information on traffic on wireless mesh networks using the IP protocol, which includes both TCP and UDP. Specifically, they employed the K-means algorithm and the Auto Class tool to categorize and distinguish between several types of session data. After aggregating the log data, we describe a technique for analyzing network traffic based on race-driven NS-2 simulations using a mixed wavelet model. The conventional Poisson model is contrasted with the more modern mixed wavelet (WM) model. As a result of their research, they determined that mixed WM provides superior performance when analyzing long-range dependence in wireless mesh networks. For the purpose of categorizing network traffic, K-means, Auto class, and DBSCAN were proposed as clustering algorithms by Jeffrey [9]. The authors collected packet traces from

ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

two sources: the Auckland IV trace and the university of Calgary trace. To compare the efficacy of the Kmeans, Auto class, and DBSCAN algorithms, they were put to use in a traffic classification scenario. We give a thorough evaluation of Kmeans, DBSCAN, and Auto class algorithms, among others, and draw some interesting conclusions. Based on the findings, they determined that Auto Class was the superior method for classifying network traffic.

K-means clustering is a data mining approach used for intrusion detection, and it was used by [10]. Different sets of data were gathered at regular intervals from the MIB network. The authors used information entropy to determine which characteristics within a massive dataset were the most important. They settled on four characteristics: the ports at both ends of an IP address, as well as the IP addresses of the respective destinations. To test the network, they introduced a variety of attacks. Data mining using K-means clustering was used to categorize the samples as either normal or malignant. They arrived at this verdict after seeing the method's effective performance in selecting superior feature qualities and a high accuracy detection rate [11].

### *3.6 Performance Evaluation*

To begin with, the accuracy of the algorithm must be measured. This can be done by comparing the results of the algorithm to those of a manual analysis. If the results match, the accuracy of the algorithm can be considered to be high. Additionally, the algorithm should also be tested with a variety of different datasets to ensure that its performance remains consistent. The speed of the algorithm is also important, as it should be able to process large amounts of data quickly. This can be measured by testing the algorithm on a variety of datasets and measuring the time it takes to complete the analysis.

The speed of the algorithm should be compared to other algorithms for the same task to ensure that it is performing as expected. The scalability of the algorithm must also be assessed, as it should be able to process large amounts of data without any decrease in performance. This can be tested by increasing the size of the dataset and measuring the algorithm's performance. Finally, the algorithm should be tested on real-world datasets to ensure that it produces valid results. The results should be compared to those of manual analysis to ensure that the algorithm is performing as expected.

In summary, the performance of the algorithm can be evaluated using a variety of measures and techniques. Accuracy, speed, scalability, and real-world testing should all be used to assess its performance. If the algorithm is performing as expected, then it can be considered to be a successful design.

## 4. Results

### *4.1 Simulation Results*

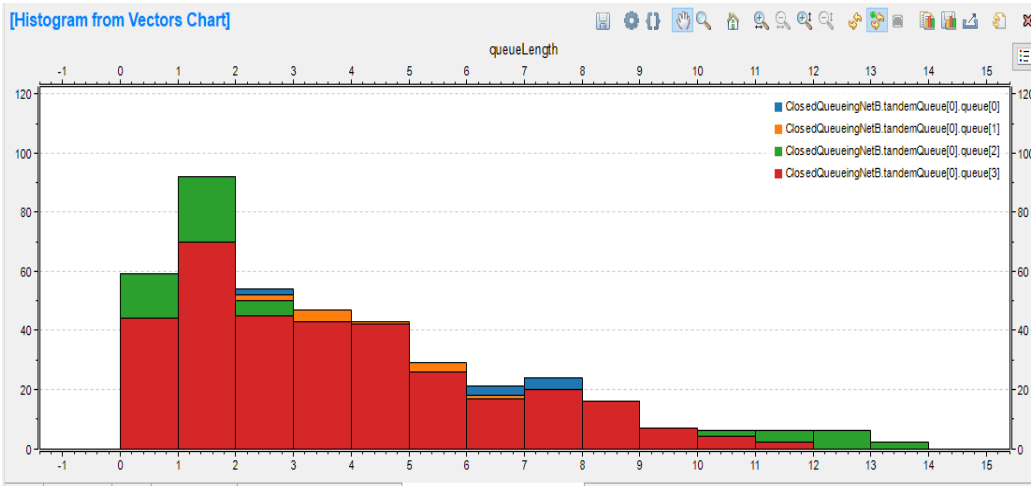


*Figure 3: Cluster Histogram*

We export all of the information from OMNET++ to graphs so that we may evaluate the outcome of "clustering the packets into tiny subsets" in our study. Packets' inter-arrival times are shown against their accuracy using simply the amount of packets in a single time period in the simulation procedure. Figure

ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

3 below displays the resulting bar plot for each threading process in the network. Each generated cluster may be further subdivided into subsets as required by the underlying application. The kNN search option of the algorithm is used to rank each cluster by the distance of its N packets to the cluster's X reference point (b).

The figure 4a and b below shows the resulting network plot during queuing process in the designed network. A network plot is a visualization technique used to represent a network of nodes and edges, where the nodes represent entities and the edges represent relationships or connections between them. In a queuing process, a network plot can be used to visualize the flow of entities (e.g. customers, requests, or jobs) through a network of queues and servers.

In a closed queuing system, the number of entities in the system is fixed, meaning that there is a finite population of entities that will be served by the system. In this context, a network plot can be used to visualize the flow of entities as they move through the system.

The nodes in the network plot represent the queues and the servers in the system. The edges represent the transitions between the nodes, which occur when an entity moves from one queue to another or from a queue to a server.At the start of the queuing process, all the entities are in the initial queue. As the entities move through the system, the network plot updates to show the current state of the system. The edges between the nodes show the flow of entities between the queues and servers.

By visualizing the network plot, we can analyze the behavior of the system and identify potential bottlenecks or areas for improvement. For example, we can identify queues that are frequently congested or servers that are underutilized. This information can be used to optimize the queuing process and improve the overall performance of the system.

The figure 4 below shows the resulting network plot during queuing process in the designed network.

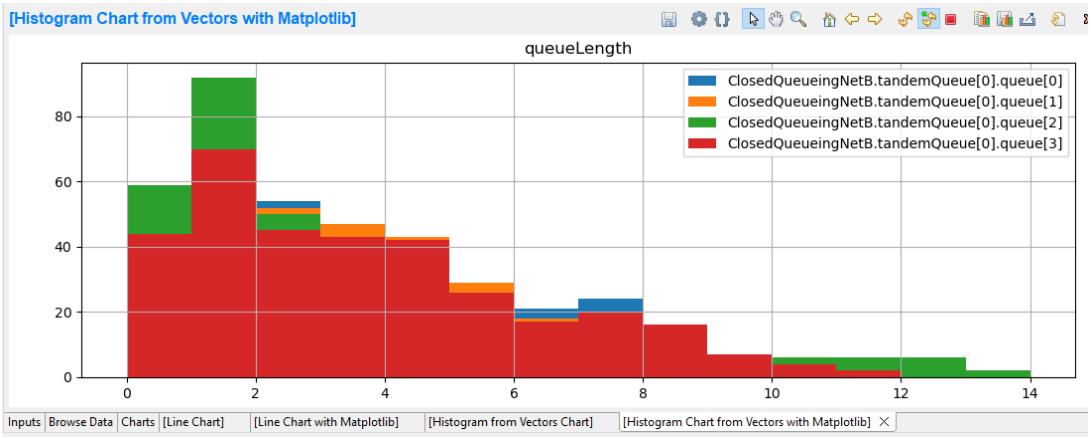


*Figure 4: Using matplotlib to plot network results*

Figure 5 shows the queuing of each threading inn the algorithm showing the best of them all

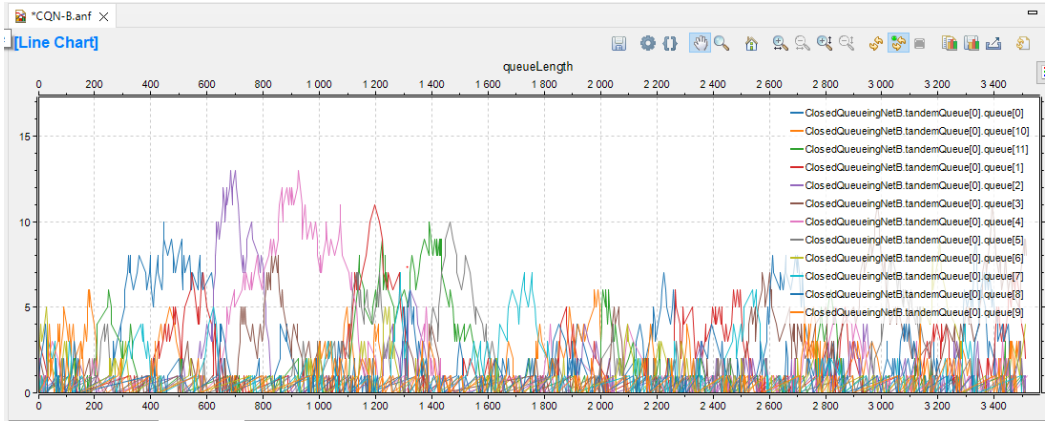


*Figure 5: Using matplotlib to plot network results*

### *4.2 Real-World Results*

In the real world analysis results the data model is shown in an example below.

ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

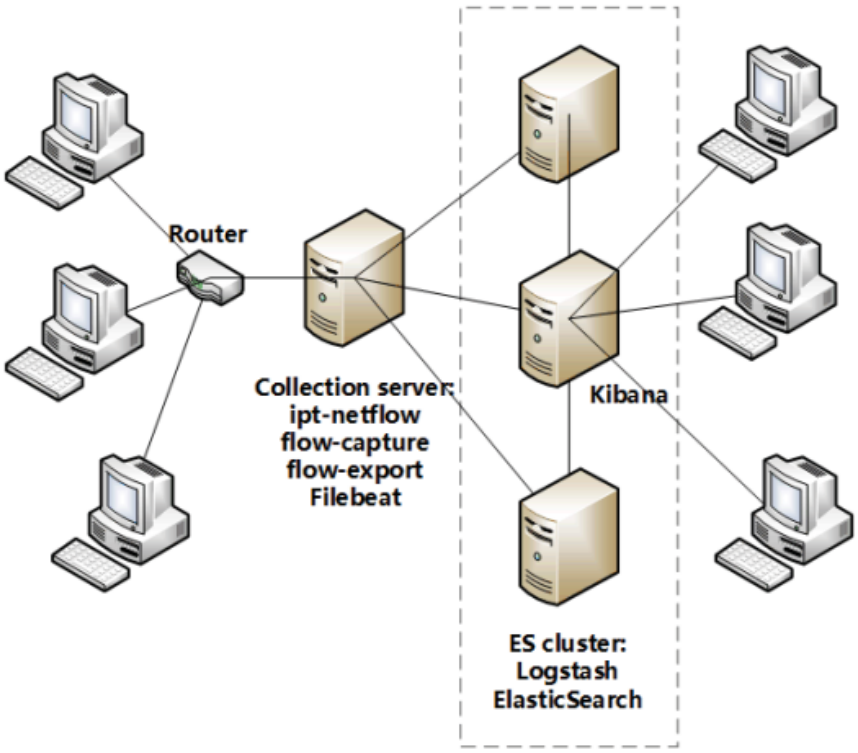


*Figure 6: System Model*

Information highway taken by each of the 256 programs. Outgoing Internet and messaging access permissions were the focus of the scan. Eighty percent or so of vendor-approved apps and eighty-four percent of unapproved applications were highlighted as suspicious in the first study, suggesting that they may be leaking sensitive information to other parties. When comparing official and unofficial apps, we found that only 26% of the red-flagged apps had access to the Internet and 42% had access to messaging. 24–25% of users for each application type had access to the Internet. Comparatively, just 34% of messages were counted while using non-official applications, while 49% were seen in official apps that provide encrypted texting. Figure 6 displays the results of these tests, showing that the official apps adhered to best practices and had their ability to connect to many devices at once carefully validated. The authors collected data on IoT device network traffic and conducted network tests.A line graph that displays the number of concurrent users in a network traffic line graph shows the number of users who are simultaneously using the network at any given time. The X-axis of the graph represents time, and the Y-axis represents the number of users. The line on the graph rises and falls to reflect the number of users who are currently active on the network.

By examining the line graph, one can determine the peak periods of network usage, as well as any patterns or trends in the number of concurrent users over time. This information can be used to optimize network performance, ensure that adequate resources are available during peak usage periods, and identify any network capacity issues that need to be addressed. It can also help in predicting and planning for future network needs based on historical usage patterns. Overall, the concurrent user line graph provides valuable insights into network usage patterns and helps network administrators make informed decisions about managing network resources.

MetricFire [12] services are advantageous because of its built-in capacity to consume time-series IoT logs and data, in addition to storing and displaying traffic linked to various patterns, loads, and dependability from IoT nodes and Cloud portals. The network traffic generated by the various IoT devices used in the experiment is seen in Figure 7, which may be found here.

In network analysis, the distribution function for outbound exfiltration refers to a statistical model used to estimate the probability distribution of data being sent out of a network, such as user data from official apps or third-party apps.

For example, the distribution function can be used to model the probability that sensitive data is being exfiltrated from a network by a malicious actor. By analyzing the outbound traffic and applying the

ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

distribution function, security analysts can identify potential exfiltration attempts and take action to prevent data breaches.

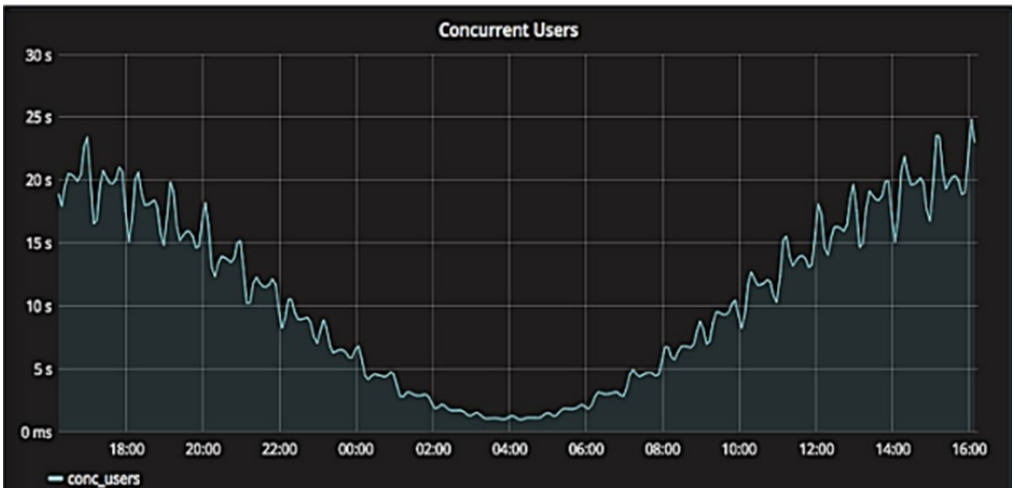


*Figure 7: Concurrent Connections of Systems*

When it comes to official apps and third-party apps, the distribution function can be used to model and compare the outbound traffic patterns of each type of app. This analysis can reveal potential security risks posed by third-party apps, as well as identify any patterns or trends in the data being sent by official apps. Overall, the distribution function for outbound exfiltration is a useful tool in network analysis for detecting and preventing data breaches, identifying potential security risks, and optimizing network performance as shown in fig 6


*Figure 8: System Model*

It was found that our experimental examination on official and third-party IoT applications was effective in identifying sensitive data flows in both of those contexts. IoT apps transmit at least five types of potentially dangerous information over the web and messaging services, as shown above in Figure 8

A possible system model for the experimental examination of official and third-party IoT applications to identify sensitive data flows is as follows:

1) Data collection: The system collects network traffic data generated by IoT applications during their normal operation, including data sent over the web and messaging services.

2) Data preprocessing: The collected data is preprocessed to extract relevant features, such as the type of data being transmitted, the destination of the data, and the frequency and volume of data transmissions.

3) Data classification: The system applies machine learning algorithms to classify the extracted features into categories based on their potential risk level. For example, sensitive data flows may be classified into a high-risk category, while non-sensitive data flows may be classified into a low-risk category.

4) Risk assessment: The system assesses the risk associated with each category of data flows based on predefined criteria, such as the type of data, the destination, and the frequency and volume of data transmissions. For example, data flows containing personal or confidential information may be deemed to be high risk, while data flows containing public information may be deemed to be low risk.

5) Reporting: The system generates reports summarizing the results of the risk assessment for both

ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

official and third-party IoT applications. The reports may include visualizations such as graphs or charts to help users understand the data flows and the associated risks.

By using this system model, the experimental examination of official and third-party IoT applications can effectively identify sensitive data flows and provide insights into the types of potentially dangerous information being transmitted over the web and messaging services. This can help organizations make informed decisions about the security of their IoT applications and take appropriate measures to protect sensitive data.

## 5. Discussion and Conclusion

### *5.1 Limitations of the Algorithm*

The limitations of the p algorithm stem from its architecture, distributed nature, and design goals, making it incompatible with certain settings.

#### *5.1.1 Minor paperwork issue*

HDFS was initially created for the purpose of media streaming.

HDFS has a default file division of 64M per block while storing data to provide data redundancy and flexibility.

Many issues arise when trying to store more than 64M files, especially if there are a big number of them.

HDFS stores files as 64M blocks, even if they're smaller.

Each file's location will be saved in the Name Node.

Because of the sheer volume of little files, the Name Node's storage capabilities will be overwhelmed.

In some cases, processing a large number of these relatively small files can even require more time and energy than processing a single large file.

#### *5.1.2 Access to processes in real time*

Hadoop isn't a good fit for low-latency data processing in real time.

For the stock market update, for instance, it is crucial that all parties involved be aware of the timeline in advance.

Also, the algorithm's inability to deal with a steady stream of data pouring into processes that must be processed instantly means that real-time/flow-based processing models cannot be implemented.

#### *5.1.3 The inability to make arbitrary changes to the file*

Hadoop's storage is particularly wasteful if you're adding or modifying files, despite its capacity for many writes per write.

#### *5.1.4 Challenges with Configuration and Optimisation*

There are more than 190 ways to tweak Hadoop's settings.

Reasonable configuration in various production scenarios, including making optimal use of available resources, is a crucial problem for ensuring the system's smooth operation.

Some of the algorithm's own techniques don't work in all contexts, while others are made more logical as part of another optimization effort.

### *5.2 Future Work*

Hadoop, a cloud computing platform developed by an open source community, has been instrumental in the success of several businesses in finding effective ways to store and handle massive amounts of data.

The algorithm's use in traffic identification and analysis offers a potential solution to the challenge of dealing with the avalanche of data generated nowadays.

ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

Existing studies have shown that the Hadoop architecture is an effective tool for storing and processing messages.

Effectiveness in handling data.

The aforementioned use cases illustrate the need of developing an environment-specific, algorithm-based system framework.

The following factors will be given careful attention in our future endeavors.

The MapReduce programming paradigm is improving in one area, and a shared memory platform is suggested.

Consequently, it will be a major area of study to figure out how to use and improve the MapReduce paradigm in parallel computing settings like distributed mobile platforms.

Current plating algorithms place more emphasis on fairness, although more efficiency is typically necessary for real-world computations.

It is still an essential path for research to look into how to develop a more efficient and fair scheduling algorithm, taking into account both the resources held by the system and the present load condition.

While HDFS and HBase can accommodate structured and unstructured data at present, the real-time demands of rapidly created big data place additional strain on the underlying access architecture.

This highlights the need of developing an access platform that can accommodate not just complex data types but also high efficiency and low latency.

Our research on trafficalgorithm classification is ongoing, and we welcome inquiries into other areas as well, including those related to the adaptation and improvement of network services, which we discuss in greater depth below. Due to the lack of a well-defined starting point for measuring traffic volumes, previous tools have experienced a wide variety of complications.

## 6. Conclusion

Over the last decade, research in several areas of computer networking has focused on better understanding and forecasting network traffic. Many scientists have worked to perfect an algorithm for analyzing and forecasting network traffic, and this method has been put into practice. In this study, we provide a literature review of research into traffic analysis in networks. We compiled and examined the many methods offered for analyzing and forecasting network traffic, such as data mining, neural networks, principal components analysis, linear and nonlinear time series models, and more. By compiling all of the publications into a single table, this study aims to provide a snapshot of the state of the art in network traffic analysis and forecasting.

The research works are categorized based on their data sets, methods, and metrics for evaluating their outcomes. A review article like this would help newcomers to the field learn more about the issue at hand. As part of the work that we have done, we have assessed a clustering technique known as IATP, which stands for "Inter-arrival time and precision," with the assistance of the OMNET++ scheduler. This was done in order to classify network traffic. Combining this metric with the inter-arrival time and accuracy allowed for a more accurate separation of a very limited number of distinct clusters, which formed the basis of the study. While our simulation results show that our implementation of a variety of flow attributes is effective at classifying packets, this level of accuracy cannot be expected in real time traffic classifiers due to the constraints under which they operate. The typical range of accuracy for real-time traffic applications is 80%-95%, however this varies widely across different uses. Better methods for spotting apps using traffic data in real time will need more research and more heuristics.

## References

*[1] Al-Naffan K., Al-Hussein M. A., Alghamdi A. S., Haque M. A., & Ahmad I. (2012). Intrusion detection using PCA based modular neural network. International Journal of Machine Learning and Computing, 2(5), 583.*
*[2] Chabaa S., Zeroual A., & Antari J. (2009, November). ANFIS method for forecasting internet traffic time series. In 2009 Mediterrannean microwave symposium (mms) (pp. 1-4).*
*[3] Datti R., & Verma B. (2010). Feature reduction for intrusion detection using linear discriminant*

ISSN 2616-5775 Vol. 6, Issue 5: 56-67, DOI: 10.25236/AJCIS.2023.060508

*analysis. International Journal on Engineering Science and Technology, 2(4), 1072-1078.*
*[4] Sivakumar R., Kumar E. A., & Sivaradje G. (2011, July). Prediction of traffic load in wireless network using time series model. In 2011 International Conference on Process Automation, Control and Computing (pp. 1-6).*
*[5] Vijayakumar M., & Parvathi R. M. S. (2010). Concept mining of high volume data streams in network traffic using hierarchical clustering. European Journal of Scientific Research, 39(2), 234-242.*
*[6] R. M. Vijayakumar, Concept Mining of High Volume Data Streams in Network Traffic Using Hierarchical Clustering, Academic Journal, 2010.*
*[7] Muda Z., Yassin W., Sulaiman M. N., & Udzir N. I. (2011, July). Intrusion detection based on K-Means clustering and Naïve Bayes classification. In 2011 7th international conference on information technology in Asia (pp. 1-6).*
*[8] Yu B., & Fei H. (2008, October). Performance impact of wireless mesh networks with mining traffic patterns. In 2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery (Vol. 2, pp. 493-497).*
*[9] Erman J., Arlitt M., & Mahanti A. (2006, September). Traffic classification using clustering algorithms. In Proceedings of the 2006 SIGCOMM workshop on Mining network data (pp. 281-286).*
*[10] Du X., Yang Y., & Kang X. (2008, December). Research of applying information entropy and clustering technique on network traffic analysis. In 2008 International Conference on Computational Intelligence and Security (Vol. 2, pp. 472-476).*
*[11] Datti R., & Verma B. (2010). Feature reduction for intrusion detection using linear discriminant analysis. International Journal on Engineering Science and Technology, 2(4), 1072-1078.*
*[12] Sadek R. A., Soliman M. S., & Elsayed H. S. (2013). Effective anomaly intrusion detection system based on neural network with indicator variable and rough set reduction. International Journal of Computer Science Issues (IJCSI), 10(6), 227.*