ISSN: 2321-2152

IJMECE

International Journal of modern electronics and communication engineering

E-Mail
editor.ijmece@gmail.com
editor@ijmece.com

www.ijmece.com

ISSN2321-2152www.ijmece .com
Vol 10, Issue.4 Nov 2022

NETWORK TRAFFIC IDENTIFICATION BASED ON MACHINE LEARNING AND DEEP PACKET INSPECTION

MOUNIKA BEDADAM[1] , BHAVYASRI GAYAM[2] , KEERTHANA PASUPULETI[3] , BHAVYA PENTYALA[4]
G.SRINIVASA RAO[5]
UG students[1,2,3,4] Associate Professor[5]

ABSTRACT: *Accurate network traffic identification is an important basis for network traffic monitoring and data analysis and is the key to improving the quality of user service. In this paper, through the analysis of two network traffic identification methods based on machine learning and deep packet inspection, a network traffic identification method based on machine learning and deep packet inspection is proposed. This method uses deep packet inspection technology to identify most network traffic, reduces the workload that needs to be identified by the machine learning method, and deep packet inspection can identify specific application traffic, and improves the accuracy of identification. The machine learning method is used to assist in identifying network traffic with encryption, new applications, and unknown features, which makes up for the disadvantage of deep packet inspection that cannot identify new applications and encrypted traffic. Experiments show that this method can improve the identification rate of network traffic.*

I. INTRODUCTION

As networking technology advances rapidly, customers' expectations for network speeds and quality continue to rise. Therefore, it has become one of the challenges in network operation and maintenance management to effectively manage and control different types of network business traffic, differentiate between services, offer varying levels of quality assurance, and cater to the business requirements of users.

ANURAG ENGINEERING COLLEGE
AUTONOMOUS
(Affiliated to JNTU-Hyderabad,Aprroved by AICTE-New Delhi)
ANANTHAGIRI (V) (M), SURYAPETA (D), TELANGANA-508206

Application-specific traffic may be easily identified on a network using network traffic identification. Classifying, identifying, and distinguishing the application of network traffic allows for the traffic of various applications to be subdivided, allowing for the provision of individualized network services, which in turn increases both the quality of network services and user happiness.

For starters, the port-based traffic identification approach has a straightforward implementation concept and does not need complex computation and analysis. It's fast enough to satisfy the needs of identifying high-speed networks quickly. The port-based traffic identification method has been gradually phased out of use as a result of the proliferation of new network applications, particularly P2P applications, which rely on random ports and camouflage ports to conceal their network communications.

Unlike other methods, which depend on the port Settings of the programmer, the feature-field based identification approach may determine the effective fields in the load. High accuracy in detecting network activity and individual applications is achieved. By focusing on the first few carefully selected packets of network data, this approach may identify activity fast. Unfortunately, this technique can only recognize preexisting apps and not create new ones since it is dependent on the feature field of the application protocol. Furthermore, load encryption network traffic cannot be identified using this technique.

Using identification technology from the field of data mining, the machine learning identification method based on the flow statistics features is able to realize traffic identification through this technique, thereby resolving issues that the first two approaches were unable to tackle, remaining immune to port and protocol feature changes, and allowing for the discovery of previously unknown applications. However, this type of approach, which relies on machine learning for both Bayesian identification and SVM (support vector machine) identification, is unable to distinguish between different applications' needs; it also lags behind in detecting traffic because it depends on the type of multiple packet flows and is easily affected by flow length; below a certain length, the misdiagnosis rate is high. Furthermore, the accuracy of this identification approach is readily influenced by dynamic network

changes and traffic attribute sets; furthermore, the computationally demanding nature of this method makes it unsuitable for real-time traffic identification of high-speed networks.

According to the principle of feature field-based identification method and flow statistics-based machine learning method, a network traffic identification method based on machine learning and DPI technology is proposed based on the analysis and comparison of the aforementioned traffic identification methods.

II. LITERATURE SURVEY

Shi Dong, Zhou Ding, Ding Wei Abstract: Network traffic identification is one of the hot research fields for network management and network security. This paper describes the current situation and common methods of network traffic identification; at the same time, this paper also states the currently popular Machine learning methods. We compared and evaluated the supervised and unsupervised classification and clustering algorithms. Introduction: Network traffic identification is an important application research direction for network management and measurement, the current network traffic identification methods roughly can be classified into four categories. Port-based method; DPI (Deep packets inspection); host behavior method; flow-based method based on machine learning. Related Work: It has become hot research between domestic and foreign experts who take traffic identification as research direction, which proceeds distinguish, intrusion detection, traffic monitoring, billing, and, management. From the beginning of the study port-based method, this method used well-known port numbers to identify Internet traffic. Results and Discussion: In this paper, we use the routine evaluation standard for verifying the effectiveness of our identification algorithm. The effectiveness of the current flow identification algorithm has the following three concepts evaluation criteria: TA (True Positive), FP (False Positive), TN (True Negative), FN (False Negative) Conclusion: This paper has studied and analyzed the machine learning algorithm for network traffic identification. Through experiments on the classification algorithm of different datasets, comparing the classical unsupervised and supervised algorithms. By comparing several unsupervised machine learning algorithms (cluster algorithm), results show that the DBSCAN algorithm

has great potential and has more advantages than the other two kinds of algorithms in precision, the modeling time is between the K-Means method and DBSCAN method.

Zuleika Nascimento, Djamel Sadok, Stenio Fernandes Informatics Center Abstract: Considerable effort has been made by researchers in the area of network traffic classification, since the Internet grows exponentially in both traffic volume and number of protocols and applications. Task of traffic identification is a complex task due to the constantly changing Internet and an increase in encrypted data. There are several methods for classifying network traffic such as port-based and Deep Packet Inspection (DPI), but they are not effective since many applications use random ports and the payload could be encrypted. Introduction: In this context, identifying network traffic is a complex task, since access to the Internet is significantly increasing, bringing with it new users with different goals. Many P2P applications are increasingly popular and accessible, such as eMule, Ares, and BitTorrent. The user's behavior is also changing and the growth of streaming video services is notable. To bring to the expert's attention what is flowing through a network is an increasingly important activity. Related Work: Recently, some methodologies have been investigated as network traffic classification tools. To classify the network traffic, the clustering k-means algorithm is used and compared to other model-based clustering methods along with rule-based classification models. Associations were found among flow parameters for several protocols and applications, such as HTTP, Mail, SMTP, DNS and IRC. Results and Discussion: The results of accuracy on the Skype application proved to be superior in the OHM (100%) against a 94.97% rate for the HM, as well as for the other metrics (except for FP). The results for correctness (CR) have improve significantly, which clearly indicates the improvement of the model when classifying applications, with less packets being classified as unknown data. Rates of FP were better tackled by the OHM regarding both applications. Conclusion: This work proposed an Optimized Hybrid Model (OHM) based on computational intelligence techniques, consisting of a rule-based model and a self-organizing map (SOM) model, both optimized by the Firefly Algorithm (FA). since the architecture has the ability of being in a constant learning process to extract patterns from new applications or protocols.

Geza Szabo, Zoltan Turanyi, Laszlo Toka Abstract: We present an automatic application protocol signature generating framework for Deep Packet Inspection (DPI) techniques with performance evaluation. We propose to utilize algorithms from the field of bioinformatics. We also present preprocessing methods to accelerate our system. Moreover, we developed several postprocessing techniques to refine the accuracy of the results. Finally, we propose a DPI system, based on approximate string matching, and find it a viable, novel alternative for the refinement of exact string-matching algorithm's results. Introduction: In-depth understanding of the Internet traffic profile is a challenging task for researchers and a mandatory requirement for most Internet Service Providers (ISP). Deep Packet Inspection (DPI) can aid to ISPs in the profiling of networked applications. With this information ISPs may then apply different charging policies, traffic shaping and offer different quality of service guarantees to selected users or applications. Related Work: Three types of protocol signature generation methods can be found in the literature: a) worm signature generation e.g., [18, 8, 10, 16], b) spam rule generation [2] and c) application signature generation [12, 14, 17, 26]. Authors of [26] presented AutoSys which extracts multiple common substring sequences from sample flows as application signature. Results and Discussion: The case when the usual DFA is substituted with sequence alignment and motifs are used. It has the highest coverage in the function of the number of signatures. Regarding the FP coverage in Figure 5, it has similar characteristics to the M+R case and has the lowest among all methods. The motifs evaluated via ASM represent a theoretical maximum of the motif expressiveness. Conclusion: In this paper we present a general framework of an automatic application protocol signature generation for Deep Packet Inspection (DPI) techniques. The proposed framework utilizes algorithms from the field of bioinformatics. In the preprocessing phase we applied a Rabin-Karp fingerprinting-based method to filter the once-occurring substrings and a prefix tree construction method to summarize the substrings with common pre- and postfixes.

III. PROPOSED SYSTEM

A network traffic Classification model based on machine learning and DPI technology is proposed. This method uses DPI Technology to identify most network traffic and reduces

workload. The workload is reduced by machine learning methods. DPI technology can identify specific application traffic and improves the accuracy of identification and also improves the identification of network traffic rate.

Traffic Identification Results Based on DPI Algorithm Flow identification algorithm in CentOS system implementation, using Wireshark capture data in the campus local area network, then carries on the processing, only keep BitTorrent, PPStream such type of P2P traffic, and belongs to the WWW HTTP traffic, finally the flow identification method based on DPI and traffic identification method based on this model to analyze traffic data. As shown in "TABLE 1", the dpi-based identification method is significantly less sensitive to PPStream traffic than to BitTorrent traffic. This is because BitTorrent P2P file sharing software is open source and its protocol features can be easily found through analysis of its programs and application protocols. DPI technology can be used to identify the corresponding network traffic. For private commercial applications of PPStream, only the protocol features can be obtained by analyzing network packets and decompiling. The accuracy is limited to some extent,

which leads to the reduction of identification recognition rate of these traffic. To identify the network flows through the machine learning method which cannot be recognized by DPI, the traffic of BitTorrent and PPStream is judged as P2P traffic, which makes up for the deficiency of DPI recognition. As shown in "TABLE II", the traffic generated by BitTorrent and PPStream is judged to be P2P traffic. The identification method adopted in this study has

significantly improved the identification of P2P traffic like BitTorrent and PPStream by combining machine learning algorithm and DPI technology to detect network traffic, thus improving the overall identification rate of network traffic.

Protocol name	Actual flow /Byte	Identified traffic /Byte	Traffic identification rate	Actual connection number	Number of connections identified	Connection identification rate
BitTorrent	107046210	96954749	90.5%	235	240	94.5%
WWW	82945	80250	96.8%	20	20	100%
PPStream	30053060	26688744	74%	230	181	72.4%

Table. 1 Traffic Identification Results Based on DPI Algorithm

Protocol name	Actual flow /Byte	Identified traffic /Byte	Traffic identification rate	Actual connection number	Number of connections identified	Connection identification rate
P2P	130034025	109980029	87.2%	402	453	94.0%
WWW	82945	80250	96.8%	20	20	100%

Table. 2 Traffic Identification Results Based on This Algorithm

ISSN2321-2152www.ijmece .com
Vol 10, Issue.4 Nov 2022

Failing to meet non-functional requirements can result in systems that fail to satisfy user needs.

IV SYSTEM DESIGN

The system development life cycle adopted in this project is the waterfall model which is based on a sequential design process. This methodology was adopted because of the flexibility it offers to researchers in terms of the sequential approach to solving problems. This methodology certifies that each stage of the design process has been successfully completed and thus guaranteeing that all stages are completely dealt with.

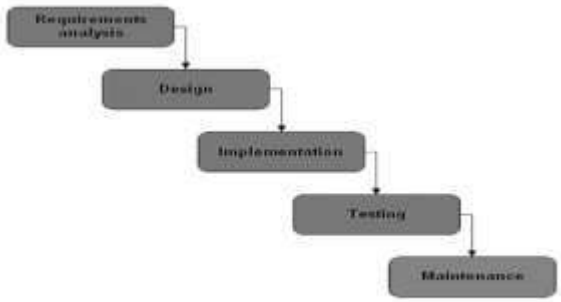

Figure 4.1: Waterfall model of the system development life cycle.

System Architecture:

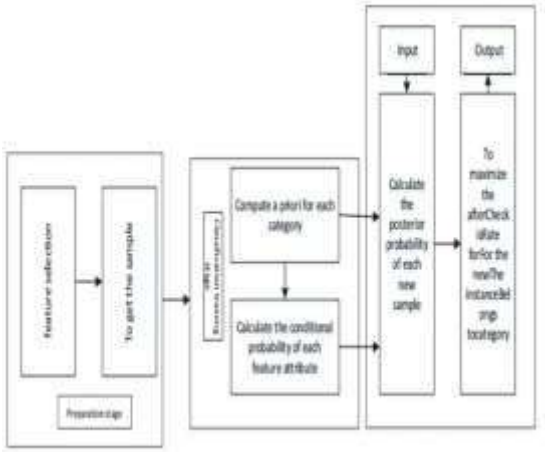

Figure 4.2: System Architecture

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users with a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.

2. Provide extensibility and specialization mechanisms to extend the core concepts.

3. Be independent of particular programming languages and development processes.

4. Provide a formal basis for understanding the modeling language.

5. Encourage the growth of the OO tools market.

6. Support higher-level development concepts such as collaborations, frameworks, patterns, and components.

7. Integrate best practices.

V SYSTEM TEST

AIM OF TESTING

The main aim of testing is to analyze the performance and to evaluate the errors that occur when the program is executed with different input sources and running in different operating environments. There are different types of approaches for testing a .NET framework-based application are;

- Unit testing
- Validation testing
- Integration testing
- User acceptance testing
- Output testing
- Black box and white box testing.

i. Unit Testing: This is the approach of taking a small part of testable application and executing it according to the requirements and testing the application behavior. Unit testing is used for detecting the defects that occur during execution (MSDN, 2010). When an algorithm is executed, the integrity should be maintained by the data structures. Unit testing is made use for testing the functionality of each algorithm during execution. Unit testing reduces the ambiguity in the units in this project, we have developed an application using different phases like encryption, and decryption. So, for getting the correct output, all the functions that are used are executed and tested.

at least once so as to make sure that all the control paths, error handling, and control structures are in a proper manner.

Limitations of Unit Testing: This is limited to testing only the functionality of the units. It can't identify integration errors, performance problems, and system problems. Unit testing can show the errors which occur in the units when the testing runs. It may not display the errors that currently are absent. ii. Validation Testing: Validation is the process of finding whether the product is built correct or not. The software application or product that is designed should fulfill the requirements and reach the expectations set by the user.

Validation is done while developing or at the final stage of development process to determine whether it is satisfies the specified requirements of user. Using validation test the developer can qualify the design, performance and its operations. Also, the accuracy, repeatability, selectivity, Limit of detection and quantification can be specified using “Validation testing” (MSDN, 2010). Output Testing: After completion of validation testing the next process is output testing. Output testing is the process of testing the output generated by the application for the specified inputs. This process checks weather the application is producing the required output as per the user’s specification or not. The “output testing” can be done by considering mainly by updating the test plans, the behavior of application with different type of inputs and with produced outputs, making the best use of the operating capacity and considering the recommendations for fixing the issues (MSDN, 2010).

iii. Integration Testing: This is an extension to unit testing, after unit testing the units are integrated with the logical program. The integration testing is the process of examining the working behavior of the particular unit after embedding with program. This procedure identifies the problems that occur during the combination of units. The integration testing can be normally done in three approaches.

- Top-down approach
- Bottom-up approach
- Umbrella approach

Testing/Predicting Stage:

In the testing/predicting stage of network traffic identification, the trained model is used to predict the protocol labels of unseen or incoming network traffic. Here's an overview of the steps involved:

1. Data Preprocessing: Similar to the training stage, the incoming network traffic data needs to be preprocessed to extract relevant features or attributes. This may involve extracting source and destination IP addresses, port numbers, packet size, time stamps, payload content, or other required information. Ensure that the preprocessing steps align with the preprocessing performed during the training stage.

2. Feature Transformation: Apply the same feature transformations and normalization techniques that were used during the training stage. This ensures consistency between the

features used during training and the features used during prediction. The transformed features should be in a suitable format to be passed into the trained model.

3. Applying the Trained Model: Pass the preprocessed and transformed network traffic data through the trained model. The model leverages the patterns and associations learned during training to predict the protocol labels for the incoming traffic. The model's output can be a single predicted label or a probability distribution over multiple possible labels, indicating the confidence or likelihood of each label.

4. Post-processing and Decision Making: Depending on the application and requirements, postprocessing steps may be performed on the predicted results. This could involve filtering, aggregating, or analyzing the predicted labels to gain insights or trigger appropriate actions based on the identified protocols. For example, the system may use the predicted labels to enforce network security policies, prioritize traffic, or trigger alerts for specific protocols.

5. Performance Evaluation: Compare the predicted protocol labels with the ground truth labels, if available, to evaluate the performance of the model during the testing/prediction stage. Performance metrics such as accuracy, precision, recall, or F1 score can be calculated to assess the model's ability to correctly identify and classify the network traffic.

6. Iterative Improvement: If the model's performance is not satisfactory, adjustments can be made. This may involve retraining the model with additional labeled data, refining the preprocessing steps, modifying the model architecture, or tuning hyperparameters to enhance the prediction accuracy.

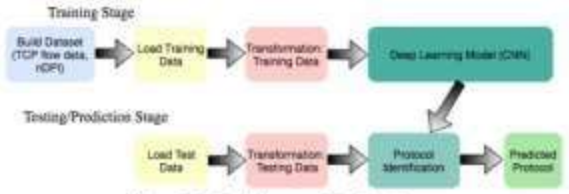

Figure 7.1: Training and Testing Stage.

VI RESULTS

Data Loading:

Figure 8.1: Data Loading

Protocol Frequency Distribution:

ISSN2321-2152www.ijmece .com
Vol 10, Issue.4 Nov 2022

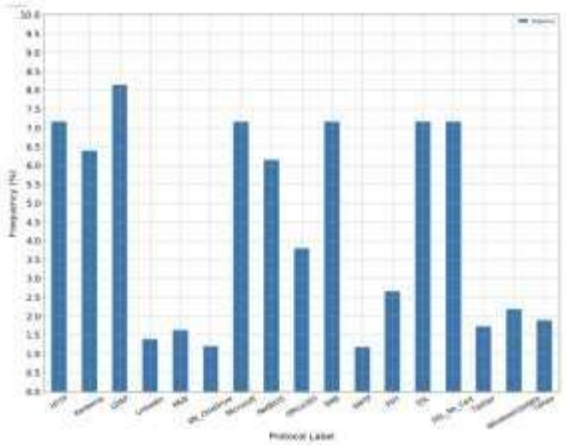

Figure 8.2: Protocol Frequency Distribution

Build a Sequential 1d-Convolutional Model:

Layer (type)	Output Shape	Param #
conv1d_2 (Conv1D)	(None, 1020, 2)	12
max_pooling1d_2 (MaxPooling1	(None, 510, 2)	0
dropout_2 (Dropout)	(None, 510, 2)	0
flatten_2 (Flatten)	(None, 1020)	0
dense_4 (Dense)	(None, 16)	16336
dense_5 (Dense)	(None, 8)	136
dense_6 (Dense)	(None, 24)	216

Total params: 16,700
Trainable params: 16,700
Non-trainable params: 0

Figure 8.3: Build a Sequential 1d-Convolutional Model

Predict: Measure the Performance of the Model on the Dataset

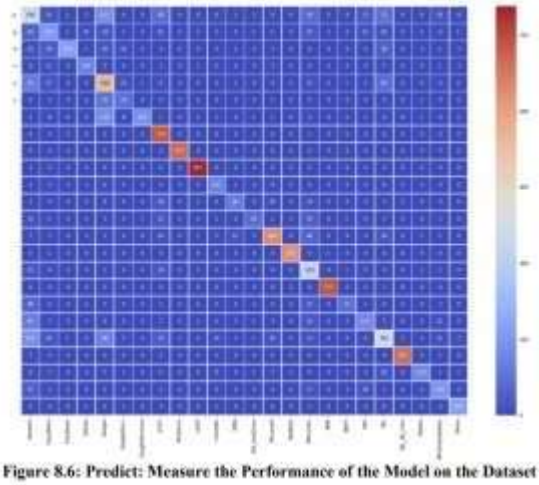

Figure 8.6: Predict: Measure the Performance of the Model on the Dataset

VII CONCLUSION

In conclusion, network traffic identification based on machine learning and deep packet inspection (DPI) is a rapidly evolving field that combines the power of advanced data analysis techniques with deep inspection of network packets. It aims to accurately classify and analyze network traffic for various purposes such as security, performance optimization, and policy enforcement. By analyzing two kinds of network traffic identification methods based on feature field and flow statistics, a network traffic identification method based on machine learning and DPI technology is proposed. This method uses DPI technology to identify most network traffic, reduces the workload that needs to be identified by the machine learning method, and improves the accuracy of identification. The machine

learning method based on the statistical characteristics of flow is used to assist the identification of network flows with encryption and unknown features, which makes up for the shortcomings of DPI technology in identifying new applications and encrypted traffic and improves the identification rate of network traffic. However, network traffic identification based on machine learning and DPI faces several challenges. These challenges include performance impact, handling encrypted traffic, dealing with protocol and application variations, addressing privacy concerns, mitigating evasion techniques, ensuring scalability, and complying with legal and regulatory requirements. To overcome these challenges and enhance network traffic identification, future advancements can focus on improving performance and scalability, effectively handling encrypted traffic, developing advanced feature extraction methods, exploring hybrid approaches, enabling real-time analysis and response, ensuring explain ability and interpretability, and building adaptive and self-learning systems.

REFERENCES

1. Kim, H., Han, S., & Lee, S. (2016). Deep packet: A novel approach to traffic classification. IEEE Communications Magazine, 54(10), 126-133.

2. Vilalta, R., & Ma, S. (2004). Learning and detecting anomalous network traffic. IEEE Intelligent Systems, 19(4), 42-49.

3.Wang, T., Li, B., & Xu, J. (2018). An overview of deep packet inspection for intrusion detection systems. Security and Communication Networks, 2018.

4.Baek, S. W., Kim, H., Han, S., Lee, S., & Hur, J. (2017). Deep packet inspection-based traffic classification using statistical flow information. IEEE Transactions on Information Forensics and Security, 12(1), 106-119.

5. Kong, Y., & Li, Q. (2019). Deep packet inspection and traffic analytics for improving network security. IEEE Communications Surveys & Tutorials, 21(1), 275-299.

6. Gao, Z., Han, S., & Kim, H. (2018). Traffic classification using deep learning: Systematic literature review. IEEE Access, 6, 23321-23334.

7. Kim, Y., & Shin, S. Y. (2020). A survey on deep learning-based network traffic identification. Sensors, 20(20), 5930.

8. Di Mauro, A., & Mazzocca, N. (2019). Machine learning techniques for traffic classification in network monitoring: A comprehensive survey. IEEE Communications Surveys & Tutorials, 21(3), 2668-2694.

9.Bernaille, L., & Teixeira, R. (2006). Traffic classification on the fly. ACM SIGCOMM Computer Communication Review, 36(2), 23-26.

10. Zhang, C., Jiang, Y., Chen, Y., & Zhang, Q. (2018). A novel network traffic identification method based on deep learning. In 2018 International Conference on Smart Internet of Things (pp. 12-16). IEEE.