2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)

Towards Aggregated Features: A Novel Proxy Detection Method Using NetFlow Data

Peipei Fu, Qingya Yang, Chang Liu, Zhenzhen Li, Gaopeng Gou, Gang Xiong, Li Guo
Institute of Information Engineering, Chinese Academy of Sciences
School of Cyber Security, University of Chinese Academy of Sciences
Beijing, China
{fupeipei, yangqingya, liuchang, lizhenzhen, gougaopeng, xionggang, guoli}@iie.ac.cn

***Abstract*—Proxies can provide privacy and anonymity protection for users, but can also be exploited by attackers to hide their malicious behaviors. In order to strengthen the monitoring and management of network, proxy detection has become an urgent and challenging task. Although a great deal of efforts has been made for proxy detection, existing methods mostly rely on the packet-level features of a single flow, such as packet inter-arrival time and payload size. In addition, handling the raw traffic throughout the communication process may lead to user privacy leakage to a certain extent. Considering the use of multi-flow statistics and reducing the invasion of user privacy, in this paper, we propose a machine learning based approach for proxy detection with NetFlow data, which only contains session-level statistical information. We extract features through NetFlow data aggregation to build machine learning model and verify the performances on different combinations of features to get the optimal feature sets. Furthermore, the importance of appropriate time windows and minimum flow numbers of NetFlow data aggregation are demonstrated by comprehensive experiments. Based on the real-world datasets, our detection method can distinguish proxy and normal traffic accurately, and achieve about 96% True Positive Rate(TPR) in our experiments with random forest classifier.**

***Index Terms*—NetFlow data, aggregated feature, proxy detection, random forest**

I. Introduction

Proxies are widely used in the current network environment. They are responsible for forwarding network traffic from clients to servers on the Internet and enable network users to access remote resources. With the development of proxy technology and the increasing demand for anonymous communication, typical proxies, such as HTTP proxy and SOCKS proxy[1], are commonly used to enable users to surf the Internet anonymously. They provide anonymity for users' identities by concealing their IP addresses and enable Internet users to bypass website censorship. Although anonymous proxies can provide protection for personal privacy and anonymity, they are also easy to be exploited by attackers to hide their malicious behaviors and identities without being caught[2].

Therefore, anonymous proxy is like a double-edged sword and brings many network security issues. In the field of network security, a great deal of researches have focused on how to detect proxy on the Internet[2]-[7]. They employed traffic flow features, timing information, payload size, digital signature, and delay measurement etc. to detect proxy or proxy user. Most of the above works made use of the packet-level information of a single flow in the raw traffic to detect proxy. Network raw traffic data often contains sensitive information about network users, which may invade users' privacy. An alternative approach that can be taken into account is to use the NetFlow data[8], which is session-level data and does not contain end-user information. Compared with the state-of-the-art methods, few studies have been found on proxy detection using NetFlow data.

NetFlow provides a session-level view of network traffic and records end-to-end connection information from the flow perspective. Therefore, it has been widely used by Internet service providers in network data collection, statistics, and analysis field. Although NetFlow data is widely available today, it also brings about some challenging problems. The main reason is that NetFlow data is collected by sampling, resulting in the inability to obtain comprehensive information. Meanwhile, the statistical attributes of the sampled data lose original representation meaning. Therefore, we adopt NetFlow data aggregation method to overcome the challenges imposed by using NetFlow data to achieve better proxy detection effect. In addition, through the approach, we aggregate statistics across multiple flows, which is not possible in a single flow.

Machine learning ideas have been applied to settle detection problems widely. In this paper, we propose to use NetFlow aggregation features along with the machine learning method to identify proxy. We focus on the aggregation feature extraction and optimization to improve the detection results. In order to obtain the optimal feature sets, we divide the aggregated features into several categories to study which combination performs best. Meanwhile, we optimize our

978-1-7281-7649-9/20/$31.00 ©2020 IEEE
DOI 10.1109/HPCC-SmartCity-DSS50907.2020.00050

aggregated features through exploring the appropriate time windows and minimum flow numbers of the NetFlow aggregation. Through experimenting with a number of machine learning methods, Random Forest is selected as our classification algorithm. Finally, we compare our method with other advanced methods to verify that our features are much better than state-of-the-art features and our method is expressive for proxy traffic detection, especially combined with Random Forest (RF) classifier, and obtains results with 0.958 TPR, 0.062 FPR, and 0.948 Accuracy.

Our contributions can be briefly summarized as follows:

- **1.** We propose a proxy detection method using a set of statistical aggregated features based on NetFlow data. NetFlow data provides session-level statistics and can effectively reduce the invasion of user privacy. To our knowledge, our study is the first attempt to detect proxy traffic based on NetFlow data, and the method can produce an excellent performance on the real-world datasets.
- **2.** The optimal feature sets that can discriminate between normal and proxy traffic are demonstrated in the paper. We experiment on different combinations of features, and discover that port, protocol, and TCP flag sequences are the main features of proxy detection instead of packet and byte sequences. Meanwhile, choosing the appropriate time windows and minimum flow numbers of NetFlow data aggregation features can effectively improve the detection result.
- **3.** Our method can produce excellent performance, outperforming state-of-the-art methods. Meanwhile, we compare and analyze different machine learning classifiers on the same dataset, our features along with Random Forest achieve the best performance in TPR, Accuracy, and FPR.

The rest of this paper is organized as follows. Firstly, we present the related work on proxy detection and NetFlow-based detection method in Section II. Section III elaborates our method in detail, which includes the process of data collection, feature extraction, feature labeling, and feature optimization. Our experiments is shown in Section IV. Finally, we conclude this paper in Section V.

II. Related Work

In the last couple of years, a lot of work has been done to investigate the proxy application and adopt novel approaches to detect proxy. In this section, we present some current related work in the area of proxy detection, as well as the previous work on NetFlow-based detection method.

A. Proxy Detection

With the development of proxy technology, it has led to the fact that network monitoring is becoming more and more difficult. Therefore, more and more attention has been paid to the proxy detection. Devin et al.[6] motivated by Nagle's algorithm, and detect the presence of a stepping stone by using the inter-arrival times and payload sizes of the packets arriving at a server. Vahid et al. [2]adopted the C4.5 classifier to identify the proxy traffic using the traffic flow features. Han et al.[7] used eight packet variables and a sequence of consecutive packet round-trip times to detect stepping stone based on neural network. These methods all depend on the assumption that the traffic generated by a regular client (without going through a proxy) would have different behaviors compared with the traffic relayed by proxy. They all propose different schemes to detect proxy using the packet-level features, e.g. the inter-arrival times and payload sizes of packets.

What's more, it also exists some new thought to detect proxy. Allen T. Webb[3] relied on network delay measurements rather than depending on the contents of the traffic to detect proxy. Han Z H et al.[5] proposed a proxy user detection method based on communication behavior portrait. Although these methods can improve the results to a certain degree, they still need to inspect the packets' information and measure the network traffic of the entire communication. Furthermore, the original traffic data often contains sensitive information about network users, which may invade users' privacy.

B. NetFlow-based Detection

As NetFlow represents a high level summary of network conversations, it can effectively avoid user privacy issues. Since NetFlow technology is made public, the protocol has been widely used in many research fields. It is extremely attractive to academic research, especially in the field of network anomaly detection[9]-[13], such as DDoS attacks, botnets, and network scan. Bilge et al.[9] presented a botnet detection system that reliably detects botnet C&C channels in readily-available NetFlow data using a set of robust statistical features. Najafabadi et al.[10] proposed an aggregation method of NetFlow data to extract the proper features for building machine learning models to detect SSH brute force attacks. Liu et al.[14] presented a method to detect network attacks and intrusions with CNN by constructing images from NetFlow data. Furthermore, NetFlow is also widely used in the field of traffic classification, identification and measurement[15]-[18]. Carela-Espanol et al.[15] studied the feasibility of identifying network applications with sampled NetFlow data using C4.5. Bakhshandeh et al.[16] proposed a method for efficiently identifying users of a network based on their behavior using the NetFlow traffic. Li et al.[18] analyzed ethereum behavior using real NetFlow data. As far as we know, no relevant research is based on NetFlow data to identify proxy traffic.

From the existing proxy detection work, we can get that most methods need to check the raw traffic, which will bring user privacy issues. Whereas NetFlow provides a high level summary of network communications, even without inspecting the contents of the packets, NetFlow performs well. Therefore, in our work, we attempt to apply NetFlow data to detect proxy based on a machine learning method, which is not from the package-level but at the session-level. Due to the statistical and sampling nature of NetFlow data, we aggregate the NetFlow data to extract features. Compared with the work in[10], our aggregation features are more abundant, and we verify which kind of aggregated features are more effective for proxy detection.

III. Methodology

In this section, our method is presented in detail. We first introduce the framework of our work. Then, we mainly introduce the process of data collection, feature extraction, feature labeling, and feature optimization.

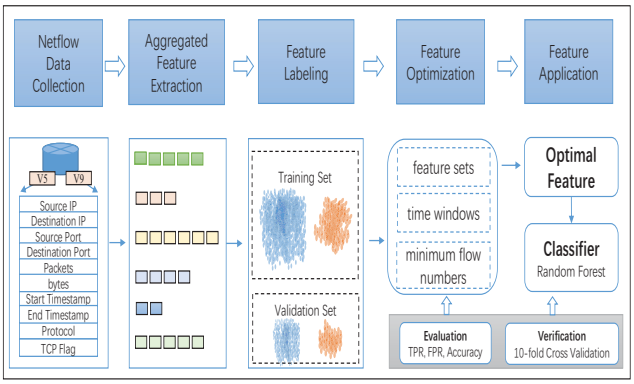

Fig. 1. The framework of our work.

A. Framework of Our Work

In the section, we mainly describe the framework of our work. In this paper, we try to use the real sampled NetFlow data to detect proxy. To study this problem, we follow the following five steps (see Figure 1), which contains NetFlow data collection, aggregated feature extraction, feature labeling, feature optimization and feature application.

First, we collect our NetFlow data from a city-level ISP network of an operator in China and analyze the original NetFlow data. After analyzing and extracting the basic information of NetFlow data (like IP, port, packets, bytes, etc.), we begin to conduct NetFlow data aggregation according to a specific key and time window. And then, abundant aggregated features in our work are extracted and labeled according to the IP information. Afterward, we begin to optimize our aggregated features in three ways, which include feature sets, time windows, and minimum flow numbers optimization. After completing the feature optimization, we apply the optimal features to the machine learning classifier (Random Forest) to detect proxy traffic. In order to evaluate our work more justly and accurately, we use 10-fold cross validation. In addition, TPR, FPR, and Accuracy are used as our assessment criteria. More details will be provided in the following.

B. Evalution Setting

In the section, we mainly describe the cross validation method, the evaluation criteria, and the classification method used in our method.

Cross Validation: In order to evaluate the reliability and stability of different methods and feature more justly and accurately, we use 10-fold cross validation. One part is kept as the test data and the other parts are used as the train data in each iteration. In order to decrease the randomness of a single result, we applied tens runs of 10-fold cross validation. The average classification results is considered as the final performance.

Evaluation criteria: In this paper, we use True Positive Rate (TPR), False Positive Rate (FPR), and accuracy (ACC) as our assessment criteria. TPR means the ratio of predicting positive samples as positive and all positive samples. FPR means the ratio of predicting negative samples as positive and all negative samples. Accuracy means the ratio of all correctly predicted samples and total samples.

Classification Method: Machine learning ideas have been applied to settle detection problems with constructed features and the advanced classification algorithms widely. In this paper, we select Random Forest[21] as our classifier. As an ensemble learning algorithm, Random Forest has excellent classification results in the field of machine learning and has strong advantages in applicability, robustness, training time, and other aspects. Therefore, Random Forest will be a good choice.

C. NetFlow Data Collection

In this section, we present how to capture NetFlow dataset. NetFlow is a network protocol proposed and implemented by Cisco Systems for traffic monitoring originally. The protocol monitors network traffic through routers or switches with no need for checking each packet, and finally summarizes network traffic into a collection of network flows. Currently, it is widely integrated into network devices, such as Cisco and Huawei. According to a survey in 2013 conducted by J. Steinberger[24] has shown that in commercial and research network operators, 70% of the participants have devices that support flow export. In addition, NetFlow supports traffic sampling, does not involve user privacy issues, greatly reduces the consumption of computing resources, so it becomes an important basic data for network traffic analysis.

The basic information of NetFlow includes: source IP, destination IP, next hop IP, source port, destination port, duration, number of packets, bytes, etc. There are several versions of NetFlow, of which NetFlow versions 5 and 9 are the most popular [19]. The V5 version is a fixed format, while V9 adopts the form of "template + record". Since NetFlow technology is made public, the NetFlow data has been widely used in flow analysis, flow reporting, threat detection and performance monitoring.

Our NetFlow data are collected from a city-level ISP network of an operator in China. The NetFlow data is sampled at a rate of 1:1000. The flows per day are about 100 million, the unique IP address is about 23 million. About 93% of these NetFlow data are version 5, and the rest are version 9. The real-time NetFlow traffic is forwarded to the servers we can reach. We adopt our high performance capture module to get the raw NetFlow traffic and analyze the traffic to collect the original NetFlow information, including source IP, destination IP, source port, destination port, start and end timestamp, protocol, TCP flag, number of packets, and number of bytes. All of these information are used to generate the final feature vector.

D. Aggregated Feature Extraction

In this part, we give a comprehensive introduction to our features. We present the data aggregation and extraction detailly.

The statistical characteristics of traffic can most intuitively reflect the network behaviors of network services. Different network services have obvious differences in statistical attributes. However, the statistical attributes of the sampled data will lose their original representation meanings. In order to deal with this problem, we design effective features from raw NetFlow data by data aggregation. In order to extract the aggregated features, we should first select the aggregation key and aggregation time window. In this paper, the IP (source IP/destination IP) is treated as a keyword, and the choice of the appropriate time window will be discussed in depth in section III-F2. Then all NetFlow records with the key IP in the time window are aggregated to generate a feature vector.

Table I shows the features extracted from aggregated NetFlow data in detail. We get 9 kinds of aggregated features, which contains NetFlow number, port sequence, packets sequence, bytes sequence etc. within aggregation. Port sequence, flag sequence and protocol sequence contain distinct number, mode number and mode frequency information within aggregation. And packets sequence, bytes sequence, duration sequence etc. contain average, standard deviation, maximum, minimum, 25 quantile, 50 quantile, 75 quantile, total, distinct, mode and mode frequency information within aggregation. Obviously, we do not use statistical values such as average and standard deviation value for "the opposite_port_sequence, TCP_flag_sequence and protocol_sequence". The main reason is that these values are meaningless for them. For example, the average number of the protocol number does not represent anything and has nothing to do with the protocol number itself. We select all these features as our feature sets and then verify the validity of the features, which will be discussed in depth in section III-F1.

E. Feature Labeling

In this section, we present how to build the ground-truth data. For machine learning algorithms, the quality of the training dataset plays a crucial role in classification accuracy. All the NetFlow data we get are not labeled with the corresponding labels. Therefore, how to get the ground-truth list is very important. In our work, we need two ground truth lists, which contain the known proxy servers and known normal servers.

The proxy server list used in our work is provided by a company that long-term engaged in proxy research. It has a profound accumulation in proxy resources. We get one week of domestic proxy nodes from their detection engine. Since proxy IP is dynamically changing, and not available every day, we set the IP that appears more than 5 days in a week as the real proxy IP in the wild. Finally, the proxy server dataset consists of 36655 unique IP addresses.

The normal server list used in our work is constructed from Alexa.cn[20]. We have used the top 1000 website domain names that are provided by Alexa. We assume that these top 1000 popular websites are not responsible for proxy service. In order to get the corresponding IP addresses, we perform DNS resolution on these domains. Through one day DNS resolution, we get 97632 IP addresses. After removing the duplicate IPs, the normal server dataset consists of 27375 unique IP addresses.

According to the above known proxy IPs and normal IPs, we begin to filter and label the NetFlow data. If the server IP in NetFlow data is consistent with the known proxy IP or normal IP, we can confirm the label of the NetFlow data. In this way, we construct our ground truth datasets to a certain degree, but still some constraints exist, such as multiple applications could be hosted on one IP.

F. Feature Optimization

In this section, we mainly describe the process of feature selection and optimization. Firstly, we verify which feature sets perform best. Then, we take the factors, time windows, and minimum flow numbers of aggregated NetFlow data, into account to further optimize our feature sets.

1) The Optimal Feature Sets Selection: In section III-D, We have collected all the possible features of NetFlow aggregation. In order to further optimization features and improve the performance, we consider to summarize all features and

TABLE I
DESCRIPTION OF AGGREGATED NETFLOW FEATURES

Feature Name	Description
num_NetFlows	Number of NetFlows/ flows within aggregation
opposite_port_sequence TCP_flag_sequence protocol_sequence	Distinct number, mode number and mode frequency within aggregation in opposite port, TCP flag and protocol.
uplink_packet_sequence downlink_packet_sequence uplink_bytes_sequence downlink_bytes_sequence flow_duration_sequence average_packet_size_sequence average_packet_interval_sequence	Average, standard deviation, maximum, minimum, 25 quantile, 50 quantile, 75 quantile, total, distinct, mode and mode frequency within aggregation in uplink packets, downlink packets, uplink bytes, downlink bytes, duration, average packet size (bytes/packets) and average packet interval (duration/packets).
Label	Class label (Proxy or Normal) associated with the NetFlows within aggregation.

TABLE II
THE DESCRIPTION OF DIFFERENT FEATURE COMBINATIONS

basic feature	num_NetFlows
A1	opposite_port_sequence TCP_flag_sequence protocol_sequence
A2	uplink/downlink_packet_sequence uplink/downlink_packet_sequence flow_duration_sequence
A3	average_packet_size_sequence average_packet_interval_sequence

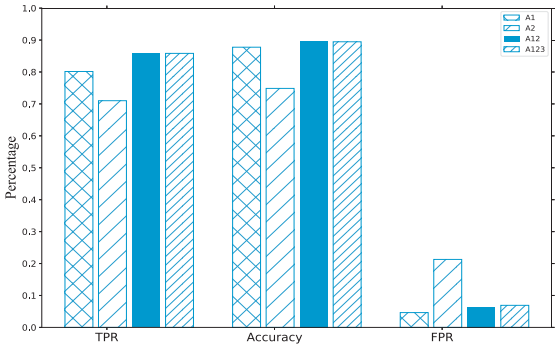

Fig. 2. TPR/FPR/Accuracy for Feature Combinations.

to find which feature combination will be the best. Therefore, all features are divided into four categories, namely basic feature, A1, A2, and A3(Table II). The basic feature contains num_NetFlows feature. Any subsequent feature combination contains basic features by default.

Then, we mainly compare the detection result of these combinations of features: A1, A2, A12, A123. Based on the comparative experiments of these combinations of features, it has been possible to verify which features are the most effective, so other combinatorial experiments are unnecessary. We evaluate the accuracy of the detection model by performing 10-fold cross-validation. All performance results are shown in Figure 2. From the result, we can get that:

(1)The result of A1 is much better than A2. This phenomenon indicates that feature combination of port, protocol, and TCP flag sequence is better than that of packets, bytes and duration sequence. Packets, bytes, and duration statistical features are not the most recognizable features in proxy detection using NetFlow data. The main reason includes two aspects: on the one hand, the total number of packets and bytes do not represent the packet sequence characteristics of the flow; on the other hand, NetFlow data is sampled, resulting in incomplete packet number and byte number at the flow level.

(2)A12 performs better than A1, which indicates that associated features have better effects than single features.

(3)The results of A12 and A123 are similar, which presents that A3 has no obvious effect on the detection effect. The most important reason is that the average packet size and average packet interval are roughly calculated value and can't convey the real information or status of packet size and interval, especially in the case of sampling.

Therefore, considering feature size and detection accuracy simultaneously, the combination of A12 will be the optimal feature sets for our method, which results in 67 dimensions totally.

2) The Optimal Time Windows Selection: The time windows of NetFlow data aggregation are very important for the performance of the detection model. As the NetFlow data is sample captured, a shorter time window might not contain enough data information about proxy service. On the other side, a larger time window will lead to long latency in detection. Najafabadi et al.[10] selected 5 minutes time window of aggregated NetFlow data, and Li et al.[18] chose 15 minutes as its time window sizes, but they chose the time windows based on their experience. Therefore, in order to

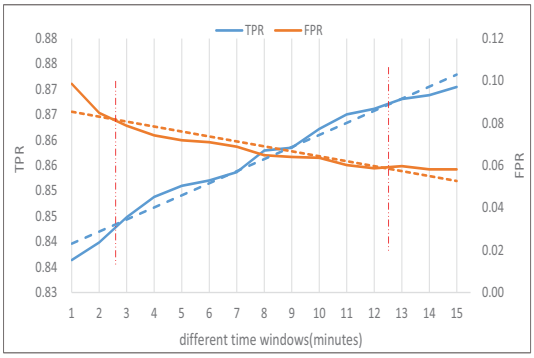

Fig. 3. Time Window Selection.

Fig. 4. Minflow Selection.

choose the appropriate time window sizes, we experiment with different time windows to evaluate the performance of the detection model.

We select the time windows from 1 minute to 15 minutes. We evaluate the accuracy of the detection model by performing 10-fold cross validation. The performance results of different time windows are presented in Figure 3. Trend lines are drawn for each type of performance line. The straight line represents the trend line, the curve represents the performance line, and the red line marks the intersection. From the experimental results, we get that the bigger the time window, the higher the TPR(Accuracy), and the lower the FPR. Although TPR(Accuracy) keeps growing, the growth trend is slower. We get that the trend line and performance line have two intersections around 3 minutes and 13 minutes. For TPR (Accuracy), the ratio between 3 minutes and 13 minutes is above the trend line. And for FPR, the ratio between 3 minutes and 13 minutes is below the trend line. The results present that the time window between 3 minutes and 13 minutes will be a better choice. In the end, considering time and accuracy simultaneously, we decide to use 8 minutes time intervals as a trade-off between the short and large time windows.

3) The Optimal Minimum Flow Numbers Selection: The work[9] discussed the influence of minimum flow numbers of aggregated features on the detection model. Therefore, we also consider to vary the minimum numbers of observed flows as an additional tunable parameter in order to improve the detection results. We use MinFlows to represent the minimum flow numbers. We evaluate eight values for MinFlows which from 5 to 40. For each experiment, we exclude the feature data which did not have more than MinFlows flows. The Figure 4 shows the detection results. The dotted lines in the figure represent the trends of TPR and Accuracy respectively.

According to the results, we can get that it's not that the more MinFlows, the better the performance. The accuracy/TPR of MinFlows from 5 to 20 is increasing continuously. When MinFlows is greater than 20, the accuracy/TPR increases slowly or even decreases. Therefore, according to the results, 20 will be the best choice for MinFlows.

IV. Experiments

Here, we list all experiments in our work. Firstly, we conduct contrast experiments to verify the superiority of our features. Then, we compare the effect of different classification algorithms on our features. In our experiments, the number of positive samples and negative samples is the same.

1) Contrast Experiments: We contrast our aggregated NetFlow features with the unoptimized aggregated features and the state-of-the-art aggregated NetFlow features.

(1) AAF, which indicates all aggregated features before optimization (listed in Table I).
(2) OAF, is our aggregated features, and the optimal feature sets are selected, time windows are set to 8 minutes and the minimum flow numbers are set to 20.
(3) EAF5, which indicates the existed aggregated NetFlow features in 5 minutes[10], which contains 19 features based upon a NetFlow's packet size, byte count, duration, etc.
(4) EAF15, namely the existed aggregated NetFlow features in 15 minutes[18], which contains 46 dimensions features about the statistical information of packets, bytes, and duration.

We apply Random Forest[21] as the classifier. The comparison results are all shown in Table III. Obviously, OAF can get 0.958 TPR, 0.062 FPR, and 0.948 accuracy on proxy detection, far better than other features. (1) EAF15(0.813 TPR) performs better than EAF5(0.717 TPR), but worse than AAF(0.847 TPR), which contains more features than EAF5 and EAF15. The results demonstrate that the more detailed and abundant the features, the better the recognition effect. (2) OAF is much better than AAF, which indicates that not all aggregation features are valid for detection and the

TABLE III
THE COMPARATIVE EXPERIMENT RESULTS BETWEEN OUR FEATURES AND OTHER EXISTING FEATURES.

Datasets	AAF			OAF			EAF5			EAF15		
	TPR	FPR	ACC	TPR	FPR	ACC	TPR	FPR	ACC	TPR	FPR	ACC
1000	0. 872	0.118	0.877	0.94	0.081	0.93	0.782	0.204	0.789	0.851	0.15	0.851
2000	0. 876	0.114	0.881	0.94	0.079	0.931	0.748	0.216	0.766	0.834	0.157	0.839
3000	0. 865	0.12	0.873	0.955	0.066	0.945	0.726	0.211	0.758	0.823	0.168	0.828
4000	0. 859	0.113	0. 873	0.963	0.059	0.952	0.708	0.213	0.748	0.808	0.168	0.82
5000	0.859	0.106	0.877	0.966	0.056	0.955	0.698	0.223	0.738	0.806	0.168	0.819
6000	0.85	0.106	0.872	0.967	0.055	0.956	0.695	0.231	0.732	0.799	0.171	0.814
7000	0.842	0.101	0.871	0.968	0.051	0.959	0.697	0.229	0.734	0.792	0.173	0.81
8000	0.835	0.097	0.869	0.968	0.05	0.959	0.684	0.233	0.725	0.789	0.177	0.806
AVE	0.847	0.109	0.874	**0.958**	**0.062**	**0.948**	0.717	0.22	0.749	0.813	0.167	0.823

choice of aggregation parameters is also very important. The comparison results present the superiority of our features in proxy detection.

2) Different Classifiers Comparison and Selection: Here, several machine learning methods, which includes Random Forest, Gradient Boosting Decision Tree(GBDT), support vector machines(SVM), decision trees(C4.5), Adaboost, Neural Networks, K-nearest-neighbors(k=5), Naive Bayes are applied to our aggregated features to detect proxy. These methods are very typical machine learning classification algorithms.

For machine learning methods, the parameters of classifier play an important role. In order to get better performance of the model, we should select appropriate parameters for the classifier. Gridsearchcv[22], which is called grid search cross validation parameter adjustment, is an effective tool for model parameter selection provided by Sklearn[23]. It performs an exhaustive search for the specified parameter value, and returns the evaluation index scores under all parameter combinations through cross validation. Therefore, we use Gridsearchcv to select the best parameters of the machine learning method. For example, in the paper, we focus on the tuning of n_estimators parameter and the max_depth parameter of Random Forest. N_estimators represents the number of decision trees in a random forest, and max_depth represents the maximum depth of decision tree. First, we choose the best parameters for n_estimators from {10-200} with a step size of 10, and other parameters are default. Then, we use the best parameters of n_estimators to optimize the parameters of max_depth from {10-200} with a step size of 10, and the other parameters are still default. Finally, when n_estimators is set to **180** and max_depth is set to **160**, we get the best performance.

Figure 5 respectively describes the experiment results of TPR, FPR and Accuracy of the eight classifiers. The results show that the Random Forest performs very well in the detection of proxy. Its TPR and Accuracy are the highest

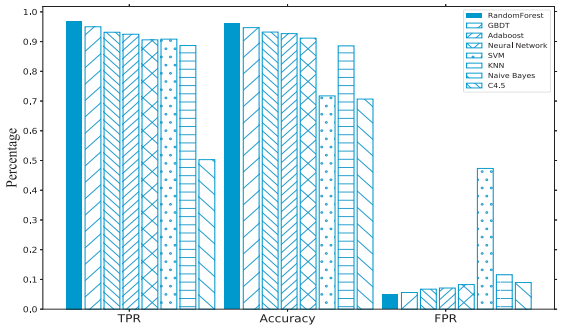

Fig. 5. The Comparative Experiment Results of Different Classifiers.

and the FPR is the lowest. Other classification algorithms are also providing good results except SVM and Naive Bayes. SVM has the highest FPR and Naive Bayes has the lowest TPR. Therefore, choosing Random Forest as the classifier is the best choice for our method. And, the results also indicate that the aggregated features extracted from NetFlow data are discriminative enough for detecting proxy.

From the above comparison experiments, we can get that our aggregated NetFlow features achieve the best performance. And our proposed method is very promising for detecting proxy. In the future, we will apply our aggregated features based on NetFlow to many other research fields. I believe the aggregated features will still work better.

3) Discussion: Our method combines aggregated NetFlow features and machine learning techniques to detect the proxy traffic. In our knowledge, this is the first time to detect proxy traffic based on sparse NetFlow data. Although we have proved that our proposed method can effectively detect the proxy traffic based on the real world NetFlow data, there are still some shortcomings that need further exploration and verification.

On one hand, our ground truth is not completely reasonable and impeccable. For NetFlow data, since it only contains flow statistics without any content information, it is difficult to label it directly. In the paper, we adopt IP as the key of the aggregated NetFlow features and use IP to label them. However, with the increasing complexity of network traffic, especially the emergence of cloud and CDN services, an IP will carry a variety of application types. Therefore, using IP alone to label the aggregated NetFlow features will lead to some inaccurate labeling. Therefore, in the future work, we need to propose a more reasonable and rigorous method to label the data to make our ground truth more credible and impeccable.

On the other hand, we have not carefully considered the unbalanced distribution of proxy traffic and normal traffic. In the actual network environment, the proxy traffic may be very small. However, in our experiments, we do not consider the class imbalance of traffic data, and take the same size of positive and negative samples. Therefore, in the future real application, the real imbalance of traffic data may lead to the degradation of our method performance. Therefore, in our next steps, we should pay more attention to the unbalanced characteristics of real Network traffic, and further improve the practicability of our method.

V. Conclusion

In this paper, we propose a novel method to detect proxy, which combines NetFlow aggregated features and Random Forest classifier. In order to solve the limitation of statistical attributes of sampling data, the paper proposes to extract features by NetFlow data aggregation. Especially, for getting the optimal feature sets, we categorize the aggregated features to verify which combinations of features work best. Furthermore, we take into account time windows and minimum flow numbers of NetFlow data to optimize our aggregated features. And, the superiority of our aggregated features is demonstrated through comprehensive comparison with the state-of-the-art features. In addition, the comparison of several typical classification algorithms is conducted, which proves that Random Forest is the best choice for our detection method. Through comprehensive optimizing and experiments, our method with Random Forest classifier can distinguish proxy and normal traffic with about 96% TPR on a real-world dataset.

VI. Acknowledgments

This work is supported by The National Key Research and Development Program of China (No.2016QY05X1000) and The National Natural Science Foundation of China (No.U1636217) and Key research and Development Program for Guangdong Province under grant No.2019B010137003. Qingya Yang is the corresponding author. Email: yangqingya@iie.ac.cn.

References

[1] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones,"SOCKS Protocol Version 5," RFC 1928, Mar. 1996.
[2] Aghaei-Foroushani, Vahid, and A. Nur Zincir-Heywood. "A proxy identifier based on patterns in traffic flows." 2015 IEEE 16th International Symposium on High Assurance Systems Engineering. IEEE, 2015.
[3] Webb, Allen T., and AL Narasima Reddy. "Finding proxy users at the service using anomaly detection." 2016 IEEE Conference on Communications and Network Security (CNS). IEEE, 2016.
[4] Dhaka, Vijaypal Singh, Vikas Lamba, and Divyanshu Pathania. "Application layer proxy detection, prevention with predicted load optimization." 2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE). IEEE, 2016.
[5] Han, Zhen-Hui, et al. "Detecting Proxy User Based on Communication Behavior Portrait." The Computer Journal 62.12, 1777-1792, 2019.
[6] Lin, Ruei-Min, Yi-Chun Chou, and Kuan-Ta Chen. "Stepping stone detection at the server side." 2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, 2011.
[7] Wu, Han-Ching, and Shou-Hsuan Stephen Huang. "Neural networks-based detection of stepping-stone intrusion." Expert Systems with Applications 37.2: 1431-1437,2010.
[8] B. Claise.Cisco systems netflow services export version 9, 2004.
[9] Bilge, Leyla, et al. "Disclosure: detecting botnet command and control servers through large-scale netflow analysis." Proceedings of the 28th Annual Computer Security Applications Conference. 2012.
[10] Najafabadi, Maryam M., et al. "Detection of ssh brute force attacks using aggregated netflow data." 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA). IEEE, 2015.
[11] Haghighat, Mohammad Hashem, and Jun Li. "Edmund: Entropy based attack detection and mitigation engine using netflow data." Proceedings of the 8th International Conference on Communication and Network Security. 2018.
[12] Terzi, Duygu Sinanc, Ramazan Terzi, and Seref Sagiroglu. "Big data analytics for network anomaly detection from netflow data." 2017 International Conference on Computer Science and Engineering (UBMK). IEEE, 2017.
[13] Hou, Jiangpan, et al. "Machine learning based ddos detection through netflow analysis." MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM). IEEE, 2018.
[14] Liu, Xiang, Ziyang Tang, and Baijian Yang. "Predicting Network Attacks with CNN by Constructing Images from NetFlow Data." 2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity). IEEE, 2019.
[15] Carela-Espanol, Valentın, Pere Barlet-Ros, and Josep Solé-Pareta. "Traffic classification with sampled netflow." traffic 33: 34-34, 2009.
[16] Bakhshandeh, Atieh, and Zahra Eskandari. "An efficient user identification approach based on Netflow analysis." 2018 15th International ISC (Iranian Society of Cryptology) Conference on Information Security and Cryptology (ISCISC). IEEE, 2018.
[17] Kiran, Mariam, and Anshuman Chhabra. "Understanding flows in high-speed scientific networks: A netflow data study." Future Generation Computer Systems 94: 72-79, 2019.
[18] Li, Zhenzhen, et al. "Ethereum Behavior Analysis with NetFlow Data." 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS). IEEE, 2019.
[19] Li, Bingdong, et al. "A survey of network flow applications." Journal of Network and Computer Applications 36.2: 567-581, 2013.
[20] Alexa Web Information Company. http://www.alexa.cn/siterank/.
[21] L. Breiman, "Random forests," Machine learning, vol. 45, no. 1, pp. 5–32, 2001.
[22] GridSearchCV,https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
[23] Sklearn, https://scikit-learn.org/stable/
[24] Steinberger, Jessica, et al. "Anomaly Detection and mitigation at Internet scale: A survey." IFIP International Conference on Autonomous Infrastructure, Management and Security. Springer, Berlin, Heidelberg, 2013.