MalFinder: An Ensemble Learning-based Framework For Malicious Traffic Detection

Candong Rong[*†], Gaopeng Gou[*†], Mingxin Cui[*†], Gang Xiong[*†✉], Zhen Li[*†], Li Guo[*†]
[*]*Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China*
[†]*School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China*
{*rongcandong, gougaopeng, cuimingxin, xionggang, lizhen, guoli*}*@iie.ac.cn*

***Abstract*—Malicious events pose a significant threat to the current increasingly interconnected Internet community. Detection based on features of network traffic and machine learning algorithms is a common approach to identify malicious events. The performance of approaches is associated with the used features and algorithms. In this paper, we propose MalFinder, an ensemble learning-based framework for malicious traffic detection. Considering the trend of network traffic encryption and the complexity of decrypting traffic, we utilize statistical features and sequence features to describe network traffic. We extend the dimensions of these two types of features to enhance their capability for representing traffic data. Feature importance analysis and contrast experiments illustrate the effectiveness of our new features. Among our selected classifiers suitable for malicious traffic detection, boosting-based classifiers XGBoost and LightGBM can reduce bias, and bagging-based classifier Random Forest can reduce variance. Stacking, which is the integration method of the classification results used in our framework, can improve the generalization ability of the method. MalFinder can achieve 96.58% F-measure and 95.44% accuracy in the malicious traffic detection task on a real-world dataset, whose results are better than those of comparison methods. In terms of unseen malicious traffic discovery, MalFinder still provides good performance with 93.46% F-measure and 91.04% accuracy, which even surpasses the results in the task of known malicious traffic detection of other comparative methods. With consideration of the scarcity of public data sets used for malicious traffic detection, we have exposed our self-built dataset for more extensive researches.**

***Index Terms*—malicious traffic detection, unseen threat discovery, ensemble learning**

I. Introduction

The issue of malicious traffic detection is a challenging problem in the field of network security. To provide a safe Internet environment for consumers, large amounts of studies focus on detecting malicious traffic [1]. Conventional signature-based methods [2] [3] only can detect attacks whose signatures are previously stored in the database. In consequence, signature-based methods have low accuracy, and they cannot detect novel attacks. Compared to signature-based methods, machine learning-based approaches can achieve better accuracy. Nowadays, machine learning-based methods are the mainstream of the approaches for detecting malicious traffic.

✉ Corresponding author.
E-mail address: xionggang@iie.ac.cn

Features extracted from network traffic and various kinds of algorithms are two main components in the machine learning-based methods. The most commonly used features in the malicious traffic detection are statistical features [4] [5], sequence features [6], and plaintext features [7] [8]. Some plaintext features are the packet content coming from HTTP, DNS, and other protocols. They are deeply influenced by the trend of network traffic encryption. Meanwhile, with the deployment of TLS 1.3 and ESNI (Encrypted Server Name Indication), the plaintext features extracted from TLS handshake messages are no longer applicable to the encrypted traffic analysis. The solutions that decrypt network traffic are computationally intensive and weaken the privacy of the users. Statistical features and sequence features are based on the statistical values of traffic data, which are straightforward to obtain. Furthermore, they are universal in different scenarios, which even can be used for fully encrypted traffic. Additionally, statistical features typically exist in the logs of network devices, and sequence features provide lots of information about the data and the relationship of each packet in the same flow. For example, the byte distribution can give information about the header-to-payload ratios, the composition of the application headers, and if any poorly implemented padding is added [6]. We start from the widely-used features, such as packet lengths, packet inter-arrival times and byte distribution, and try to dig out the statistical values of them, e.g., the maximum and minimum, for obtaining the hidden information in the data and enhancing the capability of the features for representing traffic data. Finally, we test the effectiveness of our new features in the experiments.

In terms of classifiers, how to improve the stability and accuracy of the models is the main target. The recent researches [9] [10] have illustrated meta-classifiers based on ensemble learning are often more accurate than the individual classifiers through a large number of comparative experiments. Dietterich [11] explained the reason why ensemble methods worked through theoretical analysis. In the existing studies based on ensemble learning [12] [13] [14], they did not comprehensively make use of the advantages of different implementations of ensemble learning. In our method, we select the classifiers suitable for our classification task and utilize stacking to combine boosting-based algorithms and bagging-based algorithms for improving the detection performance.

The data sets also play an essential role in the studies of malicious traffic detection. The situations of dataset usage can be summarized into four cases. The first case is that the researchers do not expose their data sets. A second case is that they use old data sets, e.g., DARPA'98 [15] and KDD'99 [16], which are considered not reflecting the current trend of network attacks. A third case is that they utilize data sets only to provide relevant feature information extracted from the traffic [17]. In the absence of raw traffic, researchers cannot do different experiments according to their requirements, limiting the use of the data sets. The last case is they employ the data sets containing the original traffic [18]. However, these data sets are typically generated in controlled environments such as sandboxes and virtual machines, and malware can evade detection in a non-executable way. Therefore, we collect traffic generated by malicious IPs, URLs, and domains existing in the real world to construct our self-built dataset. We verify the maliciousness of data and select malicious data that are still active as our data source. Considering the long time span of our data, they have caused great harm to the real world, making our dataset to be valuable.

In this paper, we present a framework based on ensemble learning, MalFinder, for malicious traffic detection and unseen threat discovery. It should be noted that malicious traffic we study and discuss in our method is web traffic generated by visiting malicious IPs, URLs, and domains. We utilize statistical features and sequence features to describe the network traffic for eliminating the impact of traffic encryption. We extend the dimensions of features with our novel statistical features and new sequence to enhance the capability of features for representing traffic data, and test the effectiveness of our features through contrast experiments. We choose suitable classifiers for malicious traffic detection. Finally, MalFinder is based on the stacking, incorporating boosting-based classifiers LightGBM and XGBoost, and bagging-based classifier Random Forest as its base classifiers and using Logistic Regression as its meta classifier. The experimental results justify that our proposed framework can successfully detect malicious traffic and discover unseen threats, with superior performance than comparative methods.

In conclusion, the main contributions of our work are briefly summarized as follows:

- We propose an ensemble learning-based framework for malicious traffic detection and unseen threat discovery. Among our selected classifiers suitable for malicious traffic detection, boosting-based classifiers XGBoost and LightGBM can reduce bias, and bagging-based classifier Random Forest can reduce variance. Stacking, which is the integration method of the classification results used in our framework, can improve the generalization ability of the model.
- We analyze the malicious network traffic and utilize statistical features and sequence features to describe the traffic while expanding the dimensions of these two types of features to enhance their capability for representing traffic data. We justify that our novel statistical features and new sequence features, and the union of them can effectively improve the corresponding detection performance of original features in contrast experiments. Considering the trend of network traffic encryption and the complexity of decrypting traffic, we do not think about features related to packet content.
- With consideration of the scarcity of public data sets used for the studies of malicious traffic detection, our dataset, which is composed of PCAP files generating by malicious IPs, URLs, and domains, has been made publicly available[1]. It contains payloads compared to the URL-based data sets.
- Our method achieves 96.58% F-measure and 95.44% accuracy on the training dataset from the real-world environment, which is higher than other ensemble learning-based methods in the comparison methods. Meanwhile, our method can achieve 93.46% F-measure and 91.04% accuracy in detecting unseen malicious traffic, which demonstrates our approach has a certain degree of migration. Furthermore, the performance is better than the results in detecting known malicious traffic of the comparative methods.

The rest of this paper is organized as follows. We review related work in Section II and describe our method in Section III. In Section IV, we discuss the experiments in detail and analyze the results. Finally, we draw the conclusion of this paper in Section V.

II. Related Work

In recent years, machine learning-based methods are widely used in researches related to malware discovery and malicious traffic detection. Prakash et al. [19] leveraged the accuracy of supervised classification on known classes and used unsupervised learning for new malware detection. Anderson et al. [8] utilized machine learning algorithms and contextual information from DNS responses and HTTP headers to identify threats in encrypted traffic.

In terms of features used in malicious traffic detection, Muhammad et al. [4] proposed two feature selection algorithms for selecting useful statistical features from flows of traffic. The Random Forest algorithm had the best performance in their experiments. Kumar et al. [5] removed UDP packets from PCAPs to concentrate only on TCP packets and selected five statistical features from raw traffic. The combination of the Random Forest algorithm and five features had the best performance. In [6], the machine learning classifiers were built using statistical features, sequence features, and features collected from the unencrypted TLS handshake messages. Hareesh et al. [7] aimed to construct a network Intrusion Detection System by creating histograms for both network header features and payloads of the packets.

Among the studies based on ensemble learning, the researchers use ensemble learning methods to improve the accuracy of tasks. For example, the bagging-based classifier Random Forest achieved good results in the tasks of [4] and

[1]https://github.com/testforexperiment/MalFinder

[5]. Sajid et al. [12] proved that the bagging ensemble learning approach using J48 as the base classifier performed better than its individual classifier for the task of spammer detection. Possebon et al. [13] combined features and ensemble learning methods to get excellent results for network traffic classification. Bijalwan et al. [14] illustrated that after using the voting method of kNN and Decision Tree, the accuracy was increased up to a higher level.

III. Methodology

We discuss the detail of our method in this section. As shown in Fig. 1, our method consists of two parts, feature selection and ensemble learning-based framework. We extract two types of features of traffic from packet capture files, namely statistical features and sequence features. The statistical features obtained through feature importance analysis are more suitable for our task, and the sequence features can reflect the relationship of different packets in the same flow. We extend the dimensions of these two types of features to enhance their capability for representing traffic data. In order to find the base classifiers for ensemble learning, we perform a preliminary screening of classifiers suitable for malicious traffic detection. We choose the top three classifiers in detection performance as the base classifiers, which are Random Forest, XGBoost, and LightGBM. As the meta classifier, Logistic Regression can further improve detection performance while avoiding over-fitting. Finally, our method has proven to be effective in detecting malicious traffic by verifying it on real-world data sets.

A. Description of Features

In this paper, we use the traffic as a bidirectional flow connection between two hosts, where the two hosts have the same 5-tuple, for instance, source and destination IP addresses, source, and destination port numbers and protocol. The forward flows are the flows sending from a client to a server, while the flows distributing from a server to clients are backward direction flows. We only focus on bidirectional flows and features in the TCP/IP protocol. Our method, combining two kinds of features, uses multiple attributes to describe the traffic. The next part describes statistical features and sequence features, respectively.

a) Statistical Features: Among the sequence features, the conventional statistical features are derived from the flow data format in IPFIX/NetFlow. We focus on the number of bytes in a data field and time interval since last packet, and then extend the set of statistical features.

- *Conventional Statistical Features:* These features include the source port and destination port; number of backward bytes, forward bytes, total bytes, backward packets, forward packets, total packets, backward packets with payload, forward packets with payload, total packets with payload; mean and standard deviation of the byte distribution; the duration of a flow in seconds.
- *Extended Statistical Features Proposed By Us:* These features are the sum, maximum, mean, median, variance, standard deviation, and covariance of the number of bytes in a data field and time interval since last packet; entropy in bits per byte and total entropy over all of the bytes in the flow; mode of the number of occurrences of packet lengths and packet inter-arrival times.
- *Feature Importance Analysis:* Finally, we find 32-dimensional features that are suitable for our classification task through feature importance analysis within Random Forest. We test the effectiveness of our new statistical features in the second part of the experiments.

b) Sequence Features: Among the sequence features, the quantile sequences are proposed by us, and the others are derived from the previous paper [6]. For the sake of simplicity, we discuss the sequences of packet lengths and packet inter-arrival times together, and describe the quantile sequences of packet lengths, packet inter-arrival times, and byte distribution together. The details are as follows:

- *The MRFTM Sequences:* A Markov Random Field transfer matrix (MRFTM) is used to model the sequence of packet lengths. We assume that the Maximum Transmission Unit (MTU) of each node in the network is 1500 bytes, and we create ten bins with 150 bytes each. The values are discretized into equally sized bins. A matrix MRFTM is then constructed where each entry, MRFTM[i, j], counts the number of transitions between the i^{th} and j^{th} bin. Finally, the rows of MRFTM are normalized to ensure a proper Markov chain. The entries of MRFTM are then used as features to the machine learning algorithms. The operation of the sequence of packet inter-arrival times is similar to the sequence of packet lengths.
- *The Quantile Sequences:* In addition to the maximum and median of the time and bytes already existing in the statistical features, we calculate the quantiles of packet lengths, packet inter-arrival times, and byte distribution to form three sequences. The three sequences have a total of 54 dimensions, each with 18 dimensions. The i^{th} percentile of values in a list can be written as follows: $P_i(\ i = 5n,\ 1 \leq n \leq 19$ and $n \neq 10)$. The second part of the experiments in the next section has illustrated the effectiveness of the quantile sequences.
- *Byte Distribution:* The byte distribution is a length-256 array that keeps a count for each byte value encountered in the payloads of the packets for each packet in the flow [6], with the n^{th} element of the array corresponding to the byte value n.

B. Ensemble Learning

There are four main ways to implement ensemble learning: boosting, bagging, stacking, and voting. Generally speaking, boosting-based methods can reduce bias, bagging-based methods can reduce variance, and stacking can improve the accuracy and generalization of models. In our method, a stacking-based layered architecture is composed of two layers, where the classifiers of the first layer are called the base classifiers, and the classifier of the second layer is called the

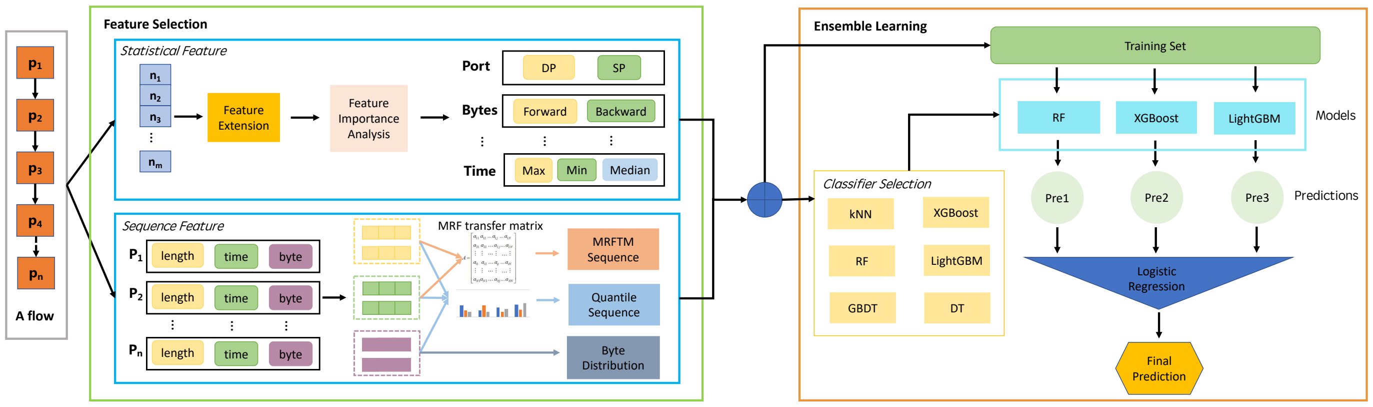

Fig. 1. The framework of our method MalFinder.

meta classifier. In the existing researches on malicious traffic detection [4] [5] [14], kNN, Decision Tree, and Random Forest have been applied to achieve good performance on their data sets. Therefore, we select these three algorithms and other ensemble learning-based algorithms as the candidate classifiers for base classifiers in the ensemble learning-based method. We use all features as the feature set and utilize implementations in Scikit-learn [20] for these algorithms. We choose the top three classifiers in detection performance, which are bagging-based algorithm Random Forest and boosting-based algorithms LightGBM and XGBoost. All features are input into different base classifiers to generate the corresponding prediction results. The predictions of the base classifiers are used to extend the original feature vector of each instance and are used as input of the meta classifier Logistic Regression. Finally, the prediction result of Logistic Regression classifier is the final label for the instance. Algorithm 1 describes the framework of ensemble learning for our solution, where the base classifiers ξ_1, ξ_2, ξ_3 correspond to XGBoost, LightGBM, Random Forest in our method, and the meta classifier ξ is Logistic Regression classifier.

Algorithm 1 Framework of ensemble learning for our solution.

```
Input:
    Training dataset D = $\{(\vec{x_1},y_1),(\vec{x_2},y_2),...,(\vec{x_m},y_m)\}$;
    Base classifiers $\xi_1,\xi_2,...,\xi_T$;
    Meta classifier $\xi$;
Output:
    $H(\vec{x}) = h^{'}(h_1(\vec{x}),h_2(\vec{x}),...,h_T(\vec{x}))$
 1: for t = 1,2,...,T do
 2:     $h_t = \xi_t(D)$;
 3: end for
 4: $D^{'} = \phi$;
 5: for i = 1,2,...,m do
 6:     for t = 1,2,...,T do
 7:         $z_{it} = h_t(\vec{x_i})$;
 8:     end for
 9:     $D^{'} = D^{'} \cup ((z_{i1},z_{i2},...,z_{iT}),y_i)$;
10: end for
11: return $h^{'} = \xi(D^{'})$;
```

IV. Experiment

In this section, we evaluate our method MalFinder on real-world data sets. Firstly, we describe the process of collecting our self-built dataset and the detail of the dataset. Secondly, we test the effectiveness of our novel statistical features and new sequence features through a set of contrast experiments. Afterwards, we justify the advantage of our ensemble learning-based method in detecting malicious traffic with superior performance over other comparative methods. Finally, we show that our method is capable of detecting unseen malicious traffic.

A. Dataset

As shown in Fig. 2, we extract useful content from DNS-BH[2] and MDL[3], and then constitute a seed dataset consisting of malicious IPs, URLs, and domains for generating malicious traffic. We process the seed dataset into PCAPs through the preprocessing module and the traffic capturer module. Firstly, the seed data is fed into the preprocessing module to test maliciousness and accessibility. We use VirusTotal[4] for differentiating whether the sample is malicious or not. VirusTotal is a comprehensive threat analysis platform. Its analysis reports are widely used in the analysis of malicious samples in academia and industry. The criteria in our solution are as follows: the sample is malicious when the result of at least one anti-virus engine is positive. At the same time, it is regarded as benign when all engines report the same negative results. Secondly, we use the traffic capturer module to catch traffic during visiting reachable IPs, URLs, and domains. Finally, we obtain the dataset TrafficData, which is composed of the PCAP files. Furthermore, we divide TrafficData into

[2]http://www.malwaredomains.com
[3]http://www.malwaredomainlist.com/mdl.php
[4]https://www.virustotal.com/

TDA and TDB, according to the date when the malicious samples first appeared in the DNS-BH and MDL data sets. The malicious samples in the TDA all appeared between December 1, 2017, and July 12, 2018, while the malicious samples in the TDB all appeared between December 11, 2018, and June 28, 2019. There is no overlap between the malicious samples in TDA and TDB, so the malicious traffic data between the two data sets are entirely different. We get benign traffic by visiting normal websites and divide it into two parts. We inject them into TDA and TDB, respectively, so that the ratio of benign traffic and malicious traffic in each part is 2:1. The benign traffic between TDA and TDB is also different. Table I illustrates a general view of our dataset TrafficData.

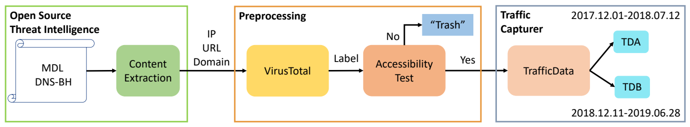

Fig. 2. The generation process of our dataset.

TABLE I
A GENERAL VIEW OF OUR DATASET TRAFFICDATA

Type		Number of PCAP Files		Number of Flows	
Label		Malicious	Benign	Malicious	Benign
TrafficData	TDA	837	1310	21245	42400
	TDB	836	1269	20067	45312
Total.		1673	2579	41312	87712

B. Analysis of Our Proposed Features

We find the statistical features which are suitable for our classification task through feature importance analysis function within Random Forest algorithm. The result is shown in Table II, and the bold parts represent our new features. We can find that among the top eight features in importance ranking, our features account for five and are ranked 2nd, 3rd, 4th, 6th, and 7th, respectively. Moreover, our features account for the vast majority among the top one-half of the features. This evidence justifies our new features are more important than those of traditional methods.

Additionally, we conduct three contrast experiments on the dataset TDA to prove the effectiveness of our novel features. Firstly, for the convenience of description, we use PRE_sta, PRE_squ, and PRE to refer to the statistical features, sequence features, and the union of them that exist in the previous methods [4] [5] [6] [14], respectively. Meanwhile, we use OUR_sta, OUR_squ, and OUR to refer to the statistical features, sequence features, and the union of them proposed by us, respectively. Secondly, all experiments utilize our ensemble learning-based method MalFinder. The corresponding results are shown in Table III. We can observe that the statistical features OUR_sta, the sequence features OUR_squ and the union of them OUR, which are proposed by us, have improved the detection performance of the corresponding original features

TABLE II
FEATURE IMPORTANCE SCORE OF STATISTICAL FEATURES

No.	Name	Score	No.	Name	Score
1	duration	0.0830	17	**byte_sum**	0.0302
2	**time_max**	0.0537	18	**byte_max**	0.0287
3	**time_median**	0.0532	19	bytes_total	0.0282
4	**time_mean**	0.0460	20	bytes_backw	0.0278
5	byte_dist_std	0.0410	21	**byte_mean**	0.0266
6	**time_cov**	0.0398	22	source port	0.0240
7	**time_std**	0.0391	23	**time_mode**	0.0212
8	byte_dist_mean	0.0384	24	num_pkts_total	0.0203
9	**total_entropy**	0.0378	25	num_pkts_backw	0.0191
10	bytes_forw	0.0378	26	**byte_mode**	0.0175
11	**time_var**	0.0376	27	pkt_count_total	0.0170
12	**entropy**	0.0376	28	num_pkts_forw	0.0158
13	**time_sum**	0.0366	29	pkt_count_forw	0.0139
14	**byte_cov**	0.0330	30	pkt_count_backw	0.0126
15	**byte_var**	0.0307	31	**byte_median**	0.0117
16	**byte_std**	0.0306	32	destination port	0.0098

to varying degrees on all indicators. The evidence has proved the effectiveness of each part of our new features as well as the whole features.

TABLE III
RESULTS OF DIFFERENT FEATURES SETS ON THE DATASET TDA

Different Feature Sets	Indicators				
	P(%)	R(%)	F1(%)	ACC(%)	AUC(%)
PRE_sta	92.24	94.47	93.34	91.02	89.31
PRE_sta+OUR_sta	**94.58**	**95.02**	**94.80**	**93.05**	**92.07**
PRE_squ	93.93	94.58	94.25	92.31	91.19
PRE_squ+OUR_squ	**95.72**	**95.29**	**95.51**	**94.03**	**93.40**
PRE	95.14	96.48	95.80	94.38	94.33
PRE+OUR	**96.62**	**96.53**	**96.58**	**95.44**	**94.91**

C. Known Malicious Traffic Detection

In the process of classifier selection, we use the grid search to adjust the hyper-parameters to optimize the parameters and use 10-fold cross-validation to ensure the stability and reliability of six candidate algorithms. Table IV illustrates an overview of the results of classifiers on the dataset TDA. We choose the top three classifiers in detection performance as the base classifiers, which are XGBoost, LightGBM, and Random Forest. These classifiers are ensemble learning methods, where XGBoost and LightGBM are boosting-based methods, and Random Forest is a bagging-based method. Based on experience, we select a simple classifier, Logistic Regression, for the meta classifier to avoid over-fitting. A two-layer stacking-based ensemble learning framework has been formed.

We compare the performance of our solution against the papers [4] [5] [14] that are most similar to our work. In the three papers, their methods have achieved good results. Muhammad et al. [4] used two feature selection algorithms to select useful features from traffic, and Kumar et al. [5] concentrate only on TCP packets and selected five features from raw traffic. The bagging-based classifier Random Forest

TABLE IV
DETECTION PERFORMANCE OF DIFFERENT CLASSIFIERS ON DATASET TDA

Name	P(%)	R(%)	F_1(%)	ACC(%)	AUC(%)
kNN	80.09	81.50	80.79	76.80	75.65
Decision Tree	91.26	90.87	91.06	88.12	86.75
GBDT	93.39	93.43	93.41	91.22	90.11
Random Forest	**94.19**	**95.17**	**94.68**	**92.88**	**91.73**
LightGBM	**96.19**	**96.06**	**96.12**	**94.84**	**94.23**
XGBoost	**96.31**	**96.17**	**96.24**	**94.99**	**94.41**

achieved the best performance in the two papers. Bijalwan et al. [14] combined kNN and decision tree classifiers by utilizing the soft voting method of ensemble learning to detect botnet traffic. The comparative experiments are performed on the dataset TDA. Table V illustrates the comparison results between our solution MalFinder and the previous methods. We can observe that our ensemble learning-based method outperforms other methods in the whole evaluation indicators, which justifies our ensemble learning-based framework is more suitable for malicious traffic detection.

In terms of time performance, since the hierarchical framework in our ensemble learning-based method uses the meta classifier Logistic Regression to process the prediction results of three classification algorithms (XGBoost, LightGBM, Random Forest) for producing the final classification results, the overall time cost is higher than other comparative methods and any single algorithm in our method.

TABLE V
RESULTS OF MALFINDER AND OTHER METHODS ON DATASET TDA

Methods	P(%)	R(%)	F_1(%)	ACC(%)	AUC(%)
Muhammad et al. [4]	71.35	71.36	71.35	61.83	57.08
Kumar et al. [5]	88.13	91.41	89.74	86.08	83.42
Bijalwan et al. [14]	91.91	91.19	91.55	88.80	87.61
MalFinder	**96.62**	**96.53**	**96.58**	**95.44**	**94.91**

D. Unseen Malicious Traffic Discovery

To reflect our method can detect unseen malicious traffic, we apply our solution MalFinder to another dataset TDB, which is entirely different from the training dataset TDA. The malicious samples in the TDA all appeared between December 1, 2017, and July 12, 2018. The malicious samples in the TDB all appeared between December 11, 2018, and June 28, 2019. The malicious traffic data in TDA and TDB are generated by entirely different samples. Table VI illustrates the performance of different methods in the task of unseen malicious traffic discovery. Even though the detection performance has decreased, our method is still better than other methods, even surpasses the results in the task of known malicious traffic detection of other methods. This benefits from the generalization ability of our proposed ensemble learning-based framework and the excellent description ability of features for traffic. The experimental results justify that our method is capable of discovering unseen threats.

TABLE VI
RESULTS OF MALFINDER AND OTHER METHODS ON DATASET TDB

Methods	P(%)	R(%)	F_1(%)	ACC(%)	AUC(%)
Muhammad et al. [4]	59.17	71.29	64.79	57.15	55.52
Kumar et al. [5]	68.34	86.52	76.36	70.5	68.7
Bijalwan et al. [14]	90.75	85.46	88.03	83.89	82.90
MalFinder	**94.55**	**92.39**	**93.46**	**91.04**	**90.19**

V. CONCLUSION

In this paper, we develop an ensemble learning-based method, MalFinder, for malicious traffic detection and unseen threat discovery. We utilize statistical features and sequence features to describe network traffic after analyzing malicious traffic data. We extend the dimensions of these two types of features to enhance their capability for representing traffic data, and demonstrate the effectiveness of our novel statistical features and new sequence features through the results of contrast experiments. Our method achieves 96.58% F-measure and 95.44% accuracy in terms of malicious traffic detection and can discover unseen malicious traffic with 93.46% F-measure and 91.04% accuracy. The experimental results show that our method MalFinder significantly outperforms other comparison methods. Furthermore, the detection performance of our method in discovering unseen malicious traffic is better than the results of comparative methods in detecting known malicious traffic. Our datasets have a total of nearly 130,000 flows. We think that achieving such good results on the given datasets reflects the effectiveness of our method. Finally, our self-built dataset has been made publicly available for the studies of malicious traffic detection. In future work, we will expand the types and scope of malicious traffic, and subdivide the malicious traffic to discover more novel malicious traffic and perform the fine-grained identification. Meantime, we will explore the possibility of applying the results of our offline analysis to the online detection for achieving its wide application in real scenarios with a lightweight implementation.

VI. ACKNOWLEDGEMENT

This work is supported by The National Natural Science Foundation of China (No.U1636217) and The National Key Research and Development Program of China (No.2016QY05X1000, No.2018YFB1800200) and Key research and Development Program for Guangdong Province under grant No.2019B010137003.

REFERENCES

[1] Buczak A L, Guven E. A survey of data mining and machine learning methods for cyber security intrusion detection[J]. IEEE Communications surveys & tutorials, 2015, 18(2): 1153-1176.

[2] Griffin K, Schneider S, Hu X, Chiueh, T. C. Automatic generation of string signatures for malware detection[C]. International workshop on recent advances in intrusion detection. Springer, Berlin, Heidelberg, 2009: 101-120.

[3] Perdisci R, Lee W, Feamster N. Behavioral Clustering of HTTP-Based Malware and Signature Generation Using Malicious Network Traces[C]. NSDI. 2010, 10: 14.

[4] Shafiq M, Yu X, Bashir A K, Chaudhry H N, Wang D. A machine learning approach for feature selection traffic detection using security analysis[J]. The Journal of Supercomputing, 2018, 74(10): 4867-4892.

[5] Kumar S, Viinikainen A, Hamalainen T. Machine learning classification model for network based intrusion detection system[C]. 2016 11th International Conference for Internet Technology and Secured Transactions (ICITST). IEEE, 2016: 242-249.

[6] Blake Anderson, Subharthi Paul, and David Mcgrew. Deciphering malware's use of tls (without decryption).Journal of Computer Virology and Hacking Techniques, 14(1):1–17, 2016.

[7] Hareesh I, Prasanna S, Vijayalakshmi M, et al. Anomaly detection system based on analysis of packet header and payload histograms[C]. 2011 International Conference on Recent Trends in Information Technology (ICRTIT). IEEE, 2011: 412-416.

[8] Anderson B, McGrew D. Identifying encrypted malware traffic with contextual flow data[C]. Proceedings of the 2016 ACM workshop on artificial intelligence and security. 2016: 35-46.

[9] Fernández-Delgado M, Cernadas E, Barro S, Amorim D. Do we need hundreds of classifiers to solve real world classification problems?[J]. The journal of machine learning research, 2014, 15(1): 3133-3181.

[10] Sagi O, Rokach L. Ensemble learning: A survey[J]. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2018, 8(4): e1249.

[11] Dietterich T G. Ensemble learning[J]. The handbook of brain theory and neural networks, 2002, 2: 110-125.

[12] Bhat S Y, Abulaish M, Mirza A A. Spammer classification using ensemble methods over structural social network features[C]. International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 02. IEEE Computer Society, 2014: 454-458.

[13] Possebon, I., da Silva, A., Granville, L., Schaeffer-Filho, A., Marnerides, A.. Improved Network Traffic Classification Using Ensemble Learning[C]. 2019 IEEE Symposium on Computers and Communications (ISCC). IEEE, 2019: 1-6.

[14] Bijalwan A, Chand N, Pilli E S, Krishna C R. Botnet analysis using ensemble classifier[J]. Perspectives in Science, 2016, 8: 502-504.

[15] DARPA 1998. Available on: https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset, February 1998.

[16] KDD Cup 1999. Available on: http://kdd.ics.uci.edu/databases/kddcup 99/kddcup99.html, Ocotber 2007.

[17] Moustafa N, Slay J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)[C]. 2015 military communications and information systems conference (MilCIS). IEEE, 2015: 1-6.

[18] Garcia S, Grill M, Stiborek J, Zunino A. An empirical comparison of botnet detection methods[J]. computers & security, 2014, 45: 100-123.

[19] Comar, P. M., Liu, L., Saha, S., Tan, P. N., Nucci, A. (2013, April). Combining supervised and unsupervised learning for zero-day malware detection. In 2013 Proceedings IEEE INFOCOM (pp. 2022-2030). IEEE.

[20] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine learning in Python[J]. Journal of machine learning research, 2011, 12(Oct): 2825-2830.