HNOP: Attack Traffic Detection Based on Hierarchical Node Hopping Features of Packets

Jinbu Geng[1,2], Zhenyu Cheng[1] ✉, Zhicheng Liu[1,3], Shuhao Li[1,2], and Rui Qin[1]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing, China
{gengjinbu, lishuhao, chengzhenyu, qinrui}@iie.ac.cn,
liuzhicheng@cert.org.cn

Abstract. Single packet attack, which is initiated by adding attack information to traffic packets, pose a great threat to cybersecurity. Existing detection methods for single packet attack just learn features directly from single packet but ignore the hierarchical relationship of packet resources, which trends to high false positive rate and poor generalization. In this paper, We conduct an extensive measurement study of the realistic traffic and find that the hierarchical relationship of resources is suitable for identifying single packet attacks. Therefore, we propose HNOP, a deep neural network model equipped with the hierarchical relationship, to detect single packet attacks from raw HTTP packets. Firstly, we construct resource node hopping structure based on the "Referer" field and the "URL" field in HTTP packets. Secondly, hopping features are extracted from the hopping structure of the resource nodes by G_BERT, which are further combined with the lexical features extracted by convolution operation from each node of the structure to form feature vectors. Finally, the extracted features are fed to a classifier, mapping the extracted features to the classification space through a fully connected network, to detect attack traffic. Experiments on the publicly available dataset CICIDS-2017 demonstrate the effectiveness of HNOP with an accuracy of 99.92% and a false positive rate of 0.12%. Furthermore, we perform extensive experiments on dataset IIE_HTTP collected from important service targets at different time. At last, it is verified that the HNOP has the least degraded performance and better generalization compared to the other models.

Keywords: Deep Learning · Malicious Traffic Detection · Hopping Features · Hierarchical Relationship.

1 Introduction

The packet is the basic unit of information transmission in TCP/IP protocol. Single packet attack (also called "atomic attack") is widely abused by attackers for information theft, causing serious losses to companies and individuals.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

According to the Akamai 2020 Security Report [1], there are 662 million web application layer attacks against the financial services sector and 7 billion against vertical businesses from 2018 to 2019. SQL injection, from all verticals, accounted for more than 72% of all attacks during this period. Therefore, it is crucial to effectively detect single packet attack.

Existing attack traffic detection methods for single packet attack are classified into flow-based and packet-based. Typically, a flow is defined as the set of packets transmitted between two IP addresses on a pair of specific ports using a specific protocol [12]. In the HTTP protocol, flow refers to the set of packets generated during the entire HTTP session. The flow-based detection [14,21] first accumulates multiple packets into a specific flow and then extracts the flow features (inflow and outflow ratio, packet time interval, stream size, number of packets per unit time, etc.), which tends to have a high accuracy. However, the demand for packet accumulation makes it have poor real-time performance. Packet-based detection [17,22] is highly efficient due to extracting traffic features (number of special characters, packet length, URL length, request method, payload information, etc.) directly from the single packet, but tends to have low accuracy due to information loss from inadequate feature extraction. Besides, the use of obfuscation techniques [10, 13, 15] makes it more difficult to extract lexical features from the attack traffic. Consequently, how to mine more effective features on packets is a crucial topic for packet-based detection.

As deep learning becomes more sophisticated, its effectiveness in extracting features is increasingly recognized by the academic community [7]. More and more attention is being paid to using deep learning methods to improve the effectiveness of feature extraction. Existing deep learning based attack traffic detection methods have made great progress in some cases, however there are two problems that need to be addressed. (1) Previous methods [19,20] almost extract lexical features from packets and neglect the hierarchical relational features, trend to high false positive rate. Hierarchical relational features, as intrinsic properties, expose the process of resource requests from the client to the server and can improve the performance of the detection method. (2) The redundant and invalid information in the packet leads to the lack of good generalization of the extracted features. Therefore, models [5, 19] based on these features are also poorly generalized for practical applications and do not meet the needs of realistic network environments.

To address the above issues, we propose a model based on hierarchical node hopping features using deep learning to detect single packet attack effectively. Our insight is built on the fact that the difference in hierarchical relationship and hopping structure due to different attack methods and purposes has great gain in identifying attack traffic. After analyzing the resource request process, we first transform the "URL" field and "Referer" field in the HTTP packet header into a hierarchical node hopping structure. Then G_BERT are utilized to extract their lexical features and hopping features respectively, which are further used for classification after being pooled. Experimental results show that our model achieves effective results in single packet attack detection.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

In summary, we make the following contributions.

- We analyze the hierarchical relationship of resources and verify its significance in detecting single packet attack traffic with ablation experiment.
- We transform the packet feature extraction problem for attack traffic into the hierarchical node hopping feature extraction, disregarding flow features and low-level protocols.
- We extract the retained hopping features from the reconstructed hierarchical node hopping structure by BERT and extract the lexical features from each node by convolution operation. Automatically extracted underlying features are difficult to modify and forge, improving the model's resistance to forgery.
- We implement a prototype of HNOP and perform a comprehensive evaluation of HNOP using CICIDS2017 [16] and IIE_HTTP datasets, demonstrating its effectiveness and generalization in detecting single packet attack.

The rest of this paper is outlined as follows. Section 2 reviews related work of HTTP attack detection. Section 3 presents our system scheme and details its different components. Section 4 elaborates the experimental results. Finally, we draw a conclusion in section 5.

2 Related Work

The work in this paper is based on hierarchical node hopping features of packets, using a method based on deep learning for attack traffic detection. In this section, we present the related research work and detection methods.

Attack traffic detection methods based on packet features are explored in many previous works. These methods are primarily based on the idea of machine learning concept of designing a set of traffic features and then modelling and training. Liu *et al.* [11] propose a Bayesian statistical model with a network traffic time-slicing feature to detect attack traffic based on the rule that network traffic changes over time, i.e., network traffic changes at different time slices and some traffic does not occur at specific time slices. Swarnkar *et al.* [17] study a Naive Bayes classifier to detect suspicious payload content in network packets. Gezer *et al.* [6] use the random forest method to monitor banking Trojans and can achieve 99.95% accuracy. Vijayanand *et al.* [18] use a genetic algorithm to select features and detect novel attack traffic by multi-support vector machines. Han *et al.* [8] combine support vector machine (SVM) and cross-entropy to detect controlled network traffic. Kabir *et al.* [9] propose a sampling and least square support vector machine (LS-SVM) for attack traffic detection, which solves the problems that SVM-based methods are ineffective and require lengthy training time when the amount of data is too large. However, such detection methods rely on feature engineering. The uncertainty and limitations of the prior knowledge prevent good robustness and generalization of the detection model.

In order to overcome the long-standing problem of feature dependence, many researchers propose methods based on deep learning to automatically extract the

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

optimal feature set from complex and redundant packets. Since features are automatically abstracted, the method can maintain good generalization as long as the data resources are satisfied [4], which is an advantage over other attack traffic detection methods. Wang *et al.* [20] use Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) to learn hierarchical spatial-temporal features. Spatial features are learned from each network packet by CNN and temporal features are drawn from the network packet sequence by RNN. Similarly, Wang *et al.* [19] use a one-dimensional convolutional layer to extract the spatial features of the original packets and a Gated Recursive Unit (GRU) structure to extract temporal features. Chiba *et al.* [3] utilize a hybrid optimization framework (IGASAA) based on Improved Genetic Algorithm (IGA) and Simulated Annealing Algorithm (SAA) to automatically build an efficient and effective Deep Neural Network (DNN) based attack traffic detection method. Geng *et al.* [5] transform the HTTP sessions into images and then extract the interactive features from these images by CNN.

To the best of our knowledge, there are little work on detecting HTTP attack traffic by leveraging the HTTP protocol. In addition, almost all previous work involves oversized packet contents and does not combine the attack characteristics of attack traffic to extract more effective features. This makes existing models, while performing well on public datasets, often perform poorly when used on realistic traffic with a time horizon. In this paper, we propose an attack traffic detection model based on hierarchical node hopping features of packets, which can get rid of the reliance on manual feature extraction and can improve the generalization performance of detecting attack traffic.

3 System Scheme

In this section, we build a hybrid structure neural network model, called HNOP. The architecture of the model is shown in Fig. 1, which is composed of hierarchical node hopping structure, G_BERT and classifier. The hierarchical node hopping structure is to transform the resource request process into a hopping problem between hierarchical nodes. G_BERT extracts features from the embedded layer and provides them to the classifier for malicious packet detection. These parts are detailed in the following sections.

3.1 Hierarchical Node Hopping Structure

The HTTP protocol defines the rules for requesting and responding to network resources between the network client and the network server. According to Occam's razor principle [2], we remove the fields in HTTP that are not linked to the resource request and keep only the "URL" field and the "Referer" field. Hence, we propose the HNOP model based on these two fields.

As shown in Fig. 2, we define the hierarchical nodes based on the entire process of the resource request. First, we define the "Referer" and "URL" fields

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

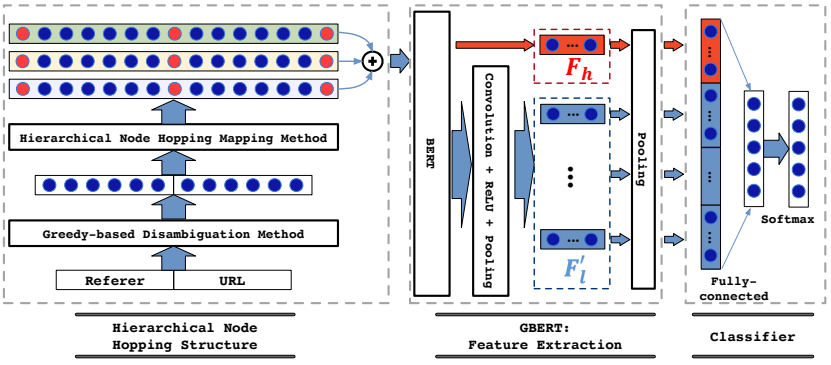


Fig. 1. The Architecture of HNOP. (F_l' : Lexical Feature; F_h : Hopping Feature).

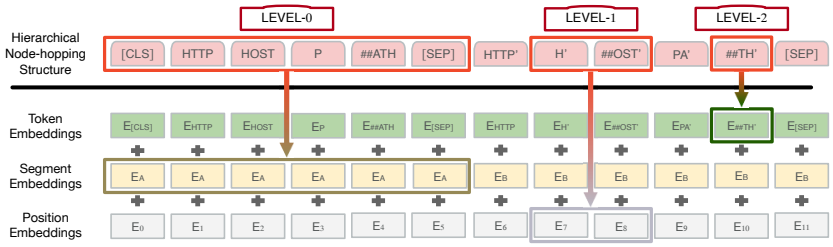


Fig. 2. Hierarchical Node Hopping Mapping.

as level-0 nodes, which characterize the page hopping relationship. Each subfield of a level-0 node is defined as a level-1 node and is used to characterize the "host-path-resource" resource request process. Finally, the subscripts of the level-1 nodes obtained by the greedy-based disambiguation method are defined as level-2 nodes for representing specific resource information. After defining hierarchical nodes, we use a hierarchical node hopping mapping method to map the hierarchical node hopping structure to the three embedding layers. Greedy-based disambiguation method and hierarchical node hopping mapping method are presented as follows.

Greedy-based Disambiguation Method The purpose of greedy-based disambiguation method is to represent the HTTP resource request process more comprehensively and completely. The greedy-based disambiguation method is shown as algorithm 1 and its steps are followed.

Step 1. Create a fixed-size vocabulary containing individual characters, common words, and sub-words that best fit the structure of the resource request.
Step 2. Combine the level-0 node "Referer" field with the "URL" field to form the original level-0 node hopping structure.
Step 3. Slice each subfield in the level-0 node into a separate level-1 node.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

Algorithm 1 Greedy-based Disambiguation Method

Input: ND $\{N_1 ... N_k\}$
Output: ND $\{w_{1_1} ... w_{1_m} ... w_{k_1} ... w_{k_l}\}$
N is the node in ND
sp is the special character
vb is the the vocabulary
f is the the greedy method-based word splitting method
for all $N \in ND$ **do**
  $longword \leftarrow N.split(sp)$
end for
for all $word \in longword$ **do**
  if $word \in vb$ **then**
    $w \leftarrow word$
  end if
  if $word \notin vb$ **then**
    $w \leftarrow f(word)[0]$
    for all $cutword \in f(word)[1:]$ **do**
      $w \leftarrow cutword.insert(0, '\#')$
    end for
  end if
end for
return ND $\{w_{1_1} ... w_{1_m} ... w_{k_1} ... w_{k_l}\}$

Step 4. Each level-1 node is firstly split into words by special characters. Words in the vocabulary are considered as independent level-2 nodes. For words not in the vocabulary, they are decomposed into sub-words and character tokens by the greedy method-based word splitting method, and each sub-word and character token is considered as a level-2 node.
Step 5. Repeat step 4 until all level-1 nodes are decomposed into level-2 nodes.

With the greedy-based disambiguation method, we get a hierarchical semantics structure. Take the "URL" (`http://ngo.mps.gov.cn/ngomh/userfiles/1/_thumbs/images/cms/article/2018/02/1-17-2banner.jpg`) as an example. "URL" and the corresponding "Referer" (`http://ngo.mps.gov.cn/ngo/portal/index.do?from=singlemessage&isappinstalled=0`) are the level-0 nodes, and the relationship between level-0 nodes is used to represent the semantic relationship of the page. The "HOST" (`ngo.mps.gov.cn`) of the "URL" is used as a level-1 node, and the relationship between level-1 nodes is used to represent the semantic relationship of order. Each subfield(`‘‘ngo’’,‘‘mps’’,‘‘gov’’,‘‘cn’’`) obtained by the greedy-based disambiguation method is used as a level-2 node. Different from the parent node, we care about the information of each level-2 node rather than the relationship between nodes.

Hierarchical Node Hopping Mapping Method Hierarchical node hopping mapping method is based on three embedding operations and is used to map the hierarchical node hopping structure to the three embedding layers. As shown

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

in Fig. 2, the nodes of each level correspond to different embedding layers. The segment embedding layer deals with level-0 nodes, maintaining the page hopping information. The position embedding layer deals with level-1 nodes, maintaining the "host-path-resource" hopping order. The token embedding layer deals with level-2 nodes, vectorizing each specific word.

The hierarchical node hopping structure of length n thus obtains three different vector representations, given as:

Token Embedding t : (1, n+3, 768), a vector representation of level-2 nodes. Note that two special nodes are inserted at the beginning ([CLS]) of the hierarchical node hopping structure and at the end ([SEP]) of each level-0 node.

Position Embedding p : (1, n+3, 768), a vector representation of the positions of the level-1 nodes.

Segment Embedding s : (1, n+3, 768), a vector representation of the positions of the level-0 nodes.

These representations are summed by element to obtain a synthetic representation:

$$X = t + p + s \tag{1}$$

Where X(1, n+3, 768) is the vectorized representation of the hierarchical node hopping structure.

3.2 G_BERT

As shown in Fig. 3, the features used in this paper are lexical features and hopping features. Lexical feature refers to the lexical feature of each level-2 node, and hopping feature refers to the association relationship between nodes at all levels. Association relationship include page hopping relationship and resource request process. That is, the hopping feature corresponds to the semantic features of level-1 nodes and level-0 nodes. In order to extract the above two features, G_BERT is divided into three layers. The upper layer uses a BERT network to extract the hopping features. The middle layer uses a convolution operation to extract lexical features. The lower layer is a feature pooling layer used to pool the features extracted from the above two layers into the final hierarchical node hopping features.

Hopping Feature Extraction Hopping features are extracted by the BERT network, which is composed of l hidden layers with the same structure (same hyperparameters) but different parameters (no shared parameters). The hidden layer includes two sub-layers, a multi-head attention function and a feed-forward function. Each sub-layer is accompanied by a residual connection.

The effect of the residual connection is to prevent the neural network gradient from disappearing or exploding in gradient. The residual connection makes the

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

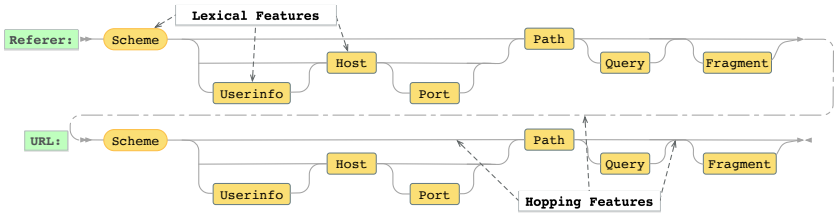


Fig. 3. Location of Lexical Feature and Hopping Feature throughout Resource Request Process. Lexical feature refers to the lexical feature of each level-2 node, and hopping feature refers to the association relationship between nodes at all levels.

loss surface smoother, which makes the model easier to train and has deeper neural network layers. With residual connectivity, the output of the sub-layer is represented as:

$$X^{i+1} = f_{ln}(X^i + (f_{sn}(X^i))) \tag{2}$$

Where X^i is the input vector, X^{i+1} is the output vector, f_{ln} is the layer normalization function for normalization along with the node embedding dimension, and f_{sn} is the current layer's operation function (multi-head attention or feed-forward).

The hopping feature is obtained by the following equation:

$$F_h = X_{CLS}^{2l+1} = X_0^{2l+1} \tag{3}$$

Where F_h is the hopping feature, X^{2l+1} is the output vector of the l th hidden layer. F_h is represented as the vector of the [CLS] node in the last layer. X_0^{2l+1} has no obvious semantic information and is more "fairly" to integrate the semantic information of each node in the input than the other X_{OTHER}^{2l+1}. Thus, it is better to represent the overall semantics of the resource request.

Lexical Feature Extraction For the node vectors N, we perform a convolution operation to extract the text features of the nodes themselves, i.e. lexical features F_l'. In this paper, F_l' are extracted as following:

$$F_l' = \sigma((F_l * K) + b) \tag{4}$$

Where K is the convolution kernel. $*$ is convolution operation, b is the bias term and σ is the ReLU activation function. After the above series of operations, the lexical features (F_l') are generated.

Hierarchical Node Hopping Feature Pooling strategy is added at the end of G_BERT, which is used for the integration of lexical features and hopping features. Three pooling strategies are adopted for comparison in this paper.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

Concat Strategy F_{lh} is obtained by splicing lexical features directly after hopping features, for F_l', F_h:

$$F_{lh} = Concat(F_h, F_l') \tag{5}$$

Max Strategy The maximum value of each dimension in F_{li}' and F_{hi} is taken to represent the feature vector F_{lh} by weighting, for F_{li}', F_{hi}:

$$F_{lhi} = Max\left(F_{hi}, F_{li}'\right) = \begin{cases} F_{hi} \ , & F_{hi} > F_{li}' \\ F_{li}' \ , & F_{hi} \leqslant F_{li}' \end{cases} \tag{6}$$

Mean Strategy Calculate the mean of F_{li}' and F_{hi} to represent the feature vector F_{lh}, for F_{li}', F_{hi}:

$$F_{lhi} = Mean\left(F_{hi}, F_{li}'\right) = \frac{1}{2}\left(F_{hi} + F_{li}'\right) \tag{7}$$

3.3 Classifier

Based on the extracted features F_{lh}, we classify the samples using fully connection layer (Linear) and SoftMax. Fully connected layer maps n eigenvectors to K (sample labeling space) eigenvectors by multiplying the weight matrix with the input vectors, adding the bias. SoftMax maps K eigenvectors to K real numbers (probabilities) of $(0, 1)$ and make sure that their sum is 1. The details are as following:

$$Y = \text{SoftMax}(z) = \text{SoftMax}\left(W_{n\times K}^T \hat{X} + b\right) \tag{8}$$

$$\hat{X} = F_{lh} \tag{9}$$

Where $\hat{X}$ is the input to the fully connected layer, $W_{n\times K}$ is the weight, b is the bias term. Based on Y, we finally obtain the class probability for each sample.

4 Evaluation

In this section, we evaluate the performance of the proposed model by performing various experiments on CICIDS2017 and IIE_HTTP. In particular, the experiments are intended to satisfy the following demands:

- Analyze the hierarchical relationship of resources and verify its significance with ablation experiment.
- Verify the effectiveness of the hierarchical node hopping feature extraction method by comparing with other methods on the publicly dataset CICIDS2017.
- Verify the generalization of HNOP on a realistic dataset IIE_HTTP by validation comparisons with other methods.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

4.1 Data Sets

Under the premise of ensuring security and data privacy, we deploy traffic collection devices at gateways on realistic main protection targets through network operators. We collect about 60 GB of attack traffic data (containing Anti_sequence _attack, SQL_injection, Webshell_attack), as well as a considerable amount of normal data in June and July 2018.

After performing the irreversible desensitization operation to protect user privacy and eliminating the threat of attack data interfering with the online environment, we obtain a long-time dataset, called IIE_HTTP. The distribution of IIE_HTTP is shown in Table 1. In deep learning, each class in the training set maintains the same number by sampling, thus avoiding model training bias.

Table 1. Reprocessing results of the IIE_HTTP.

Categories	All		Train		Test	
	Count	Percentage(%)	Count	Percentage(%)	Count	Percentage(%)
White_sample	198878	19.99	102687	18.96	96191	21.21
Anti_sequence_attack	2496	0.25	1388	0.26	1108	0.24
SQL_injection	466444	46.88	298438	55.12	168006	37.05
Webshell_attack	234195	23.54	83465	15.41	150730	33.24

4.2 The Analysis of Hierarchical Relationship

We number the level-2 nodes in the "URL" from left to right. The hopping from node n to $n+1$ is defined as out_degree of node n and in_degree of node $n+1$. We analyze the hierarchical relationship of three attack types (Anti_sequence_attack, SQL_injection, Webshell_attack).

Table 2. The hierarchical relationship features of three attack types.

Feature	**Description**	**Anti**	**SQL**	**Web**
Average node	Average value of nodes	1.42	**4.96**	4.17
Maximum node	Maximum value of nodes	5	9	**11**
Key node	Nodes with large degree	0	**10**	8
Average key node	Average value of key nodes	0	**4.36**	3.26
Associated node	Key nodes with large out and in degree	0	0	**3**
Connected graph	Amount of graphs with weak connectivity	0	**10**	3

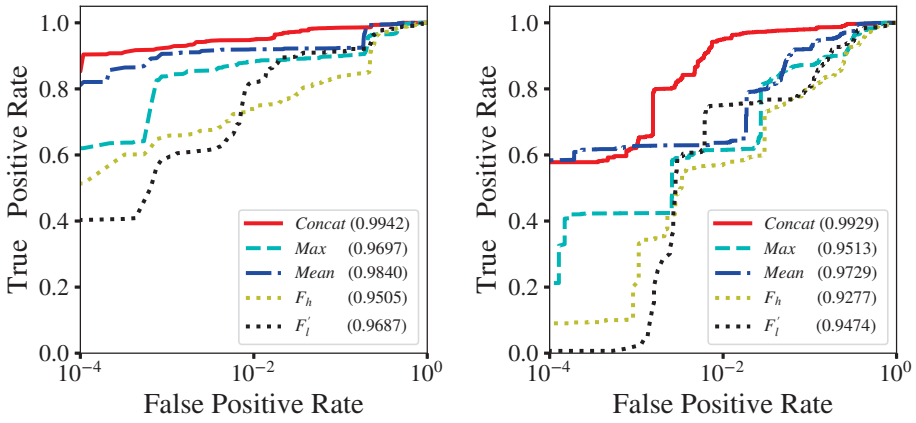
[1] *Anti: Anti_sequence_attack. SQL: SQL_injection. Web: Webshell_attack.*

As shown in Table 2, the results vary considerably. The explanation for the above phenomenon is that different attacks have different attack methods and at-

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

tack purposes. SQL_injection occurs before the intrusion and Webshell_attack occurs after the intrusion. Anti_sequence_attack mainly targets reverse sequence vulnerabilities, so its attack "path" is determined.

In our model, the hierarchical relationship are represented as lexical features and hopping features. Therefore, we conduct ablation experiments on these two features to verify the significance of hierarchical relationship in attack traffic identification. In addition, the pooling strategy is to combine lexical features and hopping features as hidden features for hierarchical relationship. we simultaneously incorporate $Concat$, Max, and $Mean$ strategies into the experiments.



(a) Macro ROC of HNOP. (b) Micro ROC of HNOP.

Fig. 4. ROC curves of HNOP. $Concat$, Max, and $Mean$ are the results of HNOP under the three pooling strategies. F_h is the result of HNOP with only hopping features, F_l' is the result of HNOP with only lexical features.

Fig. 4 show that $Concat$ has the best AUC values for both the macro ROC and the micro ROC. This demonstrates that lexical features and hopping features both play crucial roles in the model. The combination of F_l' and F_h can significantly improve the performance of HNOP compared to others using each feature alone. Note that $Mean$ and Max lose part of the feature information, causing a degradation in the performance of the model.

4.3 Compared with Other Methods

Although deep learning methods are increasingly studied for attack traffic detection in recent years, their relative proportion remains small. HNOP extracts hierarchical node hopping features from the HTTP packet level, and the neural network does this process automatically without human intervention. We expect the features to be more accurate and representative over a certain time span.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

Therefore, we compare the experimental results of our method with other published methods. There are three commonly used metrics to evaluate the performance of the model: Accuracy (ACC), Detection Rate (DR), and False Positive Rate (FPR).

Table 3. Performance comparison with other published methods on CICIDS2017 (%).

Method	INPUT	ACC	DR	FPR
Chiba *et al.* [3]	Extracted Features (70)	99.83	99.82	0.13
Geng *et al.* [5]	HTTP(header & payload)	99.07	99.01	0.40
Wang *et al.* [19]	TCP/IP(header)	99.75	99.71	0.32
HNOP	HTTP("URL" & "Referer")	**99.92**	**99.98**	**0.12**

Table 3 shows the comparison of experimental results for the dataset CICIDS2017. In this case, Our model is optimal in ACC of 99.92%, DR of 99.98% and FPR of 0.12%. Furthermore, in terms of input, we use only the "URL"and "Refer" fields of the packet header, while Wang *et al.* [19] uses all TCP/IP packet header fields. This proves that our hierarchical semantics structure in the premise of fully extracting the essential features of the packet, greatly reducing the cost of the input requirements of the detection method. Chiba *et al.*, compared to other methods, requires human extraction of features first. This not only increases the labor cost but also makes it possible for attackers to deliberately forge features due to the visibility of the features.

Table 4. Performance comparison with other published methods on IIE_HTTP (%).

Method	INPUT	ACC	DR	FPR
Chiba *et al.* [3]	Extracted Features (70)	99.22 (↓ 0.61)	98.52 (↓ 1.3)	0.53 (↑ 0.40)
Geng *et al.* [5]	HTTP(header & payload)	98.93 (↓ 0.14)	97.68 (↓ 1.33)	0.74 (↑ 0.34)
Wang *et al.* [19]	TCP/IP(header)	94.12 (↓ 5.63)	89.08 (↓ 10.63)	0.69 (↑ 0.37)
HNOP	HTTP("URL" & "Referer")	**99.84 (↓ 0.08)**	**99.71 (↓ 0.27)**	**0.11 (↓ 0.01)**

[1] *Figures in brackets indicate changes in results compared to CICIDS2017 (Table 3).*
[2] ↓ *indicate decrease,* ↑ *indicate increase.*

Table 4 presents the comparison of the experimental results for the dataset IIE_HTTP. Since the effectiveness of trained models decreases over time, all models perform slightly worse compared to the results of CICIDS2017. However, our model continues to perform well on the test set at different times, achieving ACC of 99.84%, DR of 99.71%, and FPR of 0.11%. DR and ACC drop by at least one order of magnitude less than other models. In addition, in terms of FPR, our model decreases rather than increases. This suggests that HNOP can better identify attack traffic and has higher generalization capability.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

5 Conclusion

In this paper, we propose a single packet attack traffic detection method (HNOP) that extracts the hierarchical node hopping features from the hierarchical relationship of resources. Ablation experiments fully demonstrate that hierarchical node hopping features are inherent to attack traffic learning. Extensive experiments on the public dataset demonstrate that our model achieves excellent performance with state-of-the-art work just using less traffic information. In addition, our approach outperforms the others on the realistic network dataset (IIE_HTTP) at low decay rate over time, showing higher generalization capability. In short, we present key insights on how to rereconstruct resource requests, which will shed lights on understanding single packet attack and employing proactive defenses.

Acknowledgement This work is supported by the National Key Research and Development Program of China (Grant No.2018YFB0804704), and the National Natural Science Foundation of China (Grant No.U1736218).

References

1. Akamai 2020 state of the internet/security. https://www.akamai.com/content/dam/site/en/documents/state-of-the-internet/soti-security-financial-services-hostile-takeover-attempts-report-2020.pdf, accessed October 21, 2021
2. Wikipedia. https://en.wikipedia.org/wiki/Occam%27s_razor, accessed October 21, 2021
3. Chiba, Z., Abghour, N., Moussaid, K., Rida, M., et al.: Intelligent approach to build a deep neural network based ids for cloud environment using combination of machine learning algorithms. computers & security **86**, 291–317 (2019)
4. Dong, B., Wang, X.: Comparison deep learning method to traditional methods using for network intrusion detection. In: 2016 8th IEEE International Conference on Communication Software and Networks (ICCSN). pp. 581–585. IEEE (2016)
5. Geng, J., Li, S., Zhang, Y., Liu, Z., Cheng, Z.: LIFH: Learning Interactive Features from HTTP Payload using Image Reconstruction. In: ICC 2021-IEEE International Conference on Communications. pp. 1–6. IEEE (2021)
6. Gezer, A., Warner, G., Wilson, C., Shrestha, P.: A flow-based approach for trickbot banking trojan detection. Computers & Security **84**, 179–192 (2019)
7. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 1440–1448 (2015)
8. Han, W., Xue, J., Yan, H.: Detecting anomalous traffic in the controlled network based on cross entropy and support vector machine. IET Information Security **13**(2), 109–116 (2019)
9. Kabir, E., Hu, J., Wang, H., Zhuo, G.: A novel statistical technique for intrusion detection systems. Future Generation Computer Systems **79**, 303–318 (2018)
10. Le, A., Markopoulou, A., Faloutsos, M.: Phishdef: Url names say it all. In: 2011 Proceedings IEEE INFOCOM. pp. 191–195. IEEE (2011)

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31

11. Liu, T., Qi, A., Hou, Y., Chang, X.: Method for network anomaly detection based on bayesian statistical model with time slicing. 2008 7th World Congress on Intelligent Control and Automation pp. 3359–3362 (2008)
12. Moore, A., Zuev, D., Crogan, M.: Discriminators for use in flow-based classification. Tech. rep. (2013)
13. Patil, P., Rane, R., Bhalekar, M.: Detecting spam and phishing mails using svm and obfuscation url detection algorithm. In: 2017 International Conference on Inventive Systems and Control (ICISC). pp. 1–4. IEEE (2017)
14. Pontes, C., Souza, M., Gondim, J., Bishop, M., Marotta, M.: A new method for flow-based network intrusion detection using the inverse potts model. IEEE Transactions on Network and Service Management (2021)
15. Sahoo, D., Liu, C., Hoi, S.C.: Malicious url detection using machine learning: A survey. arXiv preprint arXiv:1701.07179 (2017)
16. Stiawan, D., Idris, M.Y.B., Bamhdi, A.M., Budiarto, R., et al.: Cicids-2017 dataset feature analysis with information gain for anomaly detection. IEEE Access **8**, 132911–132921 (2020)
17. Swarnkar, M., Hubballi, N.: Ocpad: One class naive bayes classifier for payload based anomaly detection. Expert Systems with Applications **64**, 330–339 (2016)
18. Vijayanand, R., Devaraj, D., Kannapiran, B.: Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection. Computers & Security **77**, 304–314 (2018)
19. Wang, B., Su, Y., Zhang, M., Nie, J.: A deep hierarchical network for packet-level malicious traffic detection. IEEE Access **8**, 201728–201740 (2020)
20. Wang, W., Sheng, Y., Wang, J., Zeng, X., Ye, X., Huang, Y., Zhu, M.: Hast-ids: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. IEEE Access **6**, 1792–1806 (2017)
21. Xie, J., Li, S., Yun, X., Zhang, Y., Chang, P.: Hstf-model: An http-based trojan detection model via the hierarchical spatio-temporal features of traffics. Computers & Security **96**, 101923 (2020)
22. Zand, A., Vigna, G., Yan, X., Kruegel, C.: Extracting probable command and control signatures for detecting botnets. In: Proceedings of the 29th Annual ACM Symposium on Applied Computing. pp. 1657–1662 (2014)

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_31