PROCEEDINGS OF SCIENCE

Transformer-Based Detection Method for DNS Covert Channel

Qianran Sun,[a,b,*] **Junyi Liu,**[a,b] **Jiarong Wang**✉**,**[a] **Tian Yan,**[a] **Dehai An**[a] **and Fazhi Qi**[a,c]

[a] *Computing Center, Institute of High Energy Physics, Chinese Academy of Sciences,*
Beijing 100049, P.R.China

[b] *School of Nuclear Science and Technology, University of Chinese Academy of Sciences,*
Beijing 100049, P.R.China

[c] *Spallation Neutron Source Science Center,*
Dongguan 523803, P.R.China

E-mail: wangjr@ihep.ac.cn

As network technology continues to advance, network attacks against large-scale scientific facilities and science data centers have become increasingly sophisticated. The Domain Name System (DNS) protocol is a basic protocol used in the network environments of these facilities, which usually involves unencrypted data transmission to identify computers accessible through the Internet. Attackers exploit the vulnerabilities in the DNS protocol to establish covert channels, which enable them to evade traditional security detection and launch network attacks by encapsulating hidden information in DNS covert channels. These attacks can seriously compromise the network and information security of large-scale scientific facilities and science data centers. Therefore, it is imperative to detect and defend against DNS covert channels to safeguard the network of these facilities.

To address these challenges, this paper proposes a Transformer-based detection method for DNS covert channel. Our proposed method utilizes the Transformer architecture to extract global dependencies on inputs, significantly improving training speed and detection accuracy. The experimental results demonstrate that our method can provide a reliable and efficient solution for detecting DNS covert channels in large-scale scientific facilities and science data centers.

International Symposium on Grids Clouds (ISGC) 2023 in conjunction with HEPiX Spring 2023 Workshop, ISGC & HEPiX2023
19 - 31 March 2023
Academia Sinica Taipei, Taiwan

[*] Speaker

1. Introduction

Nowadays, cybersecurity has become paramount in our interconnected world. The internet is deeply embedded in various aspects of our lives, such as electronic communication, education, commercial entertainment, making it an indispensable part of contemporary people's daily routine. Unfortunately, this dependency also means that network attackers are constantly on the lookout for vulnerabilities to exploit and spread disinformation on the internet.

With the rapid expansion of computer networks and the continual emergence of various intrusion events[1], network security remains a pressing concern in the current era. Insufficient preventive measures of the network can result in data leakage, information hijacking, and even more severe consequences. Thus, prioritizing and developing network security is critical and has attracted widespread attention from governments and scholars around the world. Combating cyber threats and protecting computer network security have gradually become some of the most important and complex tasks globally[2].

In the era of the Internet, personal computers and computer networks have become primary targets for data thieves who use sophisticated techniques such as man-in-the-middle attacks or malicious software to leak data through covert channels. In these scenarios, the remote server, which typically acts as a command and control center, will wait for communication from the malicious software and record the transmitted data. To evade detection, the malicious software must find a hidden channel to leak data to remote servers, and existing security solutions may not detect or block this hidden channel[3].

Covert channels are frequently employed by attackers to conceal information within network protocols. Network covert channels, such as embedding information in redundant bits of common network protocols, like IP, TCP, ICMP, and HTTP protocols, are well-known. However, the domain name system protocol is an essential network communication protocol that is not typically blocked by firewall policies, making it an ideal secret channel for attackers[4]. DNS employs a recursive query strategy, and local DNS servers need to communicate with other DNS servers, even in a network environment under strict security policy control policies. Communication with the external network environment can still be achieved through the DNS protocol, which provides necessary conditions for the construction of network covert channels[5]. This technique has been named DNS covert channel (DCC) by the Internet Corporation for Assigned Names and Numbers (ICANN).

DNS covert channel refers to a method of transmitting information secretly by utilizing definable fields in DNS packets. These definable fields in DNS packets include the QNAME field, RDATA field, and RawUDP field. Attackers can exploit DCC as a covert channel to transmit data for malicious software, botnets, and worms. Detecting and preventing these channels is crucial to preserving the integrity and confidentiality of network data. Consequently, researchers have developed a variety of techniques for detecting DCC[6], including statistical analysis, machine learning, and signature-based methods.

In this paper, we propose a novel approach for detecting DCC based on the Transformer model, which is a powerful deep learning architecture widely used in natural language processing and computer vision tasks. Our method exploits the transformer's capability to capture complex sequential patterns and relationships, enabling it to learn and distinguish between legitimate and covert DNS traffic. By training the transformer on a large dataset of both benign and covert DNS traffic, our approach can detect DNS covert channels with high accuracy and low false positive rates. We believe that our approach will provide an effective solution for network administrators and security analysts to identify and prevent DNS covert channel attacks.

2. Related Works

In recent years, there has been a significant research interest in the detection and analysis of DNS covert channels, resulting in the proposal of many detection methods and the involvement of numerous researchers. Many machine learning-based detection methods for DCC rely on manual features, which require complex data preprocessing and feature extraction. Compared to traditional methods, deep learning-based methods can automatically extract data features without manual intervention and implement an end-to-end traffic identification model. Moreover, existing deep learning-based covert channel detection methods suffer from low recognition rates and long training periods. Existing research on DCC detection methods can be broadly categorized into three groups:

2.1 The Traditional Methods

The traditional methods for detecting DNS covert channels rely on statistical attributes such as differences in DNS packet size distribution and cross entropy. By analyzing the sub-domain entropy, it can be concluded that if a reasonable threshold can be found, DNS covert channel communication domain names can be effectively divided. However, the effectiveness of these methods in detecting encrypted information is limited, and they mainly rely on expert experience.

2.2 The Classical Machine Learning Methods

The classical machine learning methods employ traditional Bayesian techniques and utilize a combination of 12 features, including query and response packet size, as well as statistical features, for detection. To analyze DNS request messages and corresponding messages in PCAP files, these methods leverage machine learning models, such as random forest and anomaly detection[7]. Despite providing better detection results than traditional methods, their effectiveness is still limited when it comes to encrypted text.

Buczak et al.[8] proposed a machine learning method based on random forest that analyzes DNS request messages and corresponding messages in PCAP files, and extracts 16-dimensional features as input for the model. Homem et al.[9, 10] conducted a detailed analysis of subdomain entropy characteristics and achieved classification by setting entropy thresholds. Jawad Ahmed et al.[11] proposed an anomaly detection-based machine learning model that extracts features from fully qualified domain names as input for classification.

Due to the encryption involved in the transmission of information through covert channels, manual feature extraction cannot effectively extract features from encrypted information. Machine learning methods that rely on text statistical features mainly utilize information such as packet length and domain name length. Therefore, their detection efficacy is limited and mostly dependent on expert experience.

2.3 The Deep Learning Methods

The deep learning methods utilize deep neural networks to automatically learn features for detection, including communication content features from encrypted text and other parts of DNS data packets. This approach has gained popularity in recent years, as it enables the classification task to be completed using DNS text information as input. By learning the characteristics of normal DNS traffic, the deep neural networks can effectively detect DCC by calculating the mean square error between normal and malicious samples[12].

Chang Liu et al.[13] proposed a byte-level Convolutional Neural Network (CNN)[14] based method that utilizes the entire DNS packet as input to complete the classification task. Shaojie Chen et al.[15] proposed a Long Short-Term Memory (LSTM)[16] Recurrent Neural Network (RNN)[17] based method that uses fully qualified domain names as inputs for the classification task. Jiacheng Zhang et al.[18] proposed a Stacking-based method that uses DNN, 1D-CNN, and RNN network structures to detect single DNS requests, with the payload portion of DNS requests as inputs. The communication content-based detection scheme can identify single DNS traffic and has the ability to detect it in real-time. Zhang Meng et al.[19] proposed an improved Convolutional Neural Network (RDCC-CNN) based DNS covert channel recognition method that extracts effective text information from DNS packets, combines it, and converts it into grayscale images for image classification using CNN to detect DNS covert channels. Compared to directly classifying text using neural networks, there is an additional step of converting to grayscale images, which can utilize advanced techniques in the CNN field in recent years[19] to improve detection performance.

Table 1: Main papers using deep learning methods for detection

Time	2019	2020
Literature	[13]	[20]
Method	CNN	RDCC-CNN
Training Set	Iodine, Dns2tcp, Dnscat2, OzymanDNS, Reverse_DNS_hell	DNSCat, Iodine, PSUDP, Dns2tcp, tcp-over-dns
Testing set	Iodine, Dns2tcp, Dnscat2, OzymanDNS, Reverse_DNS_hell	DNSCat, Iodine, PSUDP, Dns2tcp, tcp-over-dns
Objective	The problem of binary classification detection for known datasets.	The problem of binary classification detection for known datasets.
Result	Accuracy of 99.98%, recall rate of 99.96%, F1 Score of 0.9998.	Accuracy of 99.50%, false alarm rate of 0.55%.

In summary, significant progress has been made by researchers in developing effective DNS covert channel detection methods. These methods have become crucial tools in the fight against cyber threats and are instrumental in protecting computer network security. However, with the ever-evolving nature of cyber threats, attackers are continuously developing new techniques. Therefore, it is essential to keep improving and developing new detection methods to keep up with these emerging threats.

Compared to traditional statistical methods and machine learning methods, the proposed Transformer-Based Detection Method for DNS Covert Channel in this paper has several advantages:

- Elimination of manual parameter tuning: Unlike traditional statistical methods, the proposed detection method automatically learns data features through the model, reducing the possibility of errors caused by manual intervention.
- Enhanced robustness and adaptability: The proposed detection method can adaptively learn the data's features and patterns, making it more flexible in adapting to different types and scales of datasets. Moreover, the proposed method can automatically learn the distribution of data, enhancing its robustness and making it easier to detect novel DNS covert channel attacks.
- Processing of complex sequence data: The proposed method can handle complex sequence data with long sequence dependencies, which is highly effective in detecting DNS covert channel attacks. This is in contrast to traditional machine learning methods that typically require manual feature extraction from sequence data.

Overall, the proposed Transformer-Based Detection Method for DNS Covert Channel offers a more automated approach to learning data patterns and features, thereby mitigating some of the limitations of traditional methods. This method has potential applications in detecting various types of DNS covert channel attacks.

3. Method

3.1 Data Preprocessing

Data preprocessing is a critical step in any machine learning project, and it is no exception in the case of DCC detection. The data packets collected for this task are typically in the PCAPNG format, which contains a lot of information that is irrelevant to the task at hand. Therefore, the first step in data preprocessing is to convert the PCAPNG file to a more manageable CSV format. This process involves parsing the packet headers to extract relevant information, such as source and destination IP addresses, port numbers, and protocol types.

The subsequent step is to eliminate irrelevant information items, such as serial numbers, request times, source and destination IP addresses, protocol names, and request lengths. This step is essential to reduce the dataset's size and concentrate solely on the network domain name information items that are pertinent to DCC detection.

After cleaning the data, the next crucial step is to label the dataset to distinguish whether a given entry is a DCC or not. This step is essential for supervised learning, as the model needs to be trained on labeled data to learn the patterns that differentiate DCC from normal domain names.

Table 2: Partial schematic dataset

Content	Label
www.botanicabookshop.com	0
mail.fbit.hu	0
r24070.tunnel.tuns.org	1
www.timeforkids.com	0
mail.i9-consultoria.com	0
r39845.tunnel.tuns.org	1
r30939.tunnel.tuns.org	1
flux.com	0
www.elaineespinosa.com.ghs.googlehosted.com	0
mail.jsmtech.com	0
mail.cut.ac.zw	0
r29753.tunnel.tuns.org	1
www.mypuzzledheart.com.ghs.googlehosted.com	0
r50361.tunnel.tuns.org	1
r34599.tunnel.tuns.org	1
onforb.es	0

After preprocessing and labeling the data, the next step is to train the model. The input to the model consists of the network domain name sequence and its corresponding label. To prepare the sequence for input into the model, the DistilBERT tokenizer is used to divide the sequences into tokens. The tokenizer adds special tokens for sequence classification tasks, such as [CLS] at the beginning of the sequence and [SEP] at the end of the sequence. Furthermore, the tokenizer replaces each token with the corresponding ID in the embedding table, which allows the model to learn from the sequence data.

Training the model involves feeding the preprocessed data into the model and adjusting its weights to minimize the error between the predicted output and the actual output. This process is repeated over several epochs until the model achieves high accuracy in distinguishing between DCC and normal domain names. Once the model is trained, it can be used to detect DCC in the network traffic.

3.2 Model

The Transformer model[21], which was first proposed in 2017, represents a major breakthrough in natural language processing and has formed the basis for several state-of-the-art models such as the Generative Pre-Training (GPT) and the Bidirectional Encoder Representation from Transformers (BERT) models. The Transformer is a powerful deep learning model that leverages the self-attention mechanism to differentially weigh the importance of each input component. The Transformer model adopts an encoder-decoder architecture, incorporating the attention mechanism inside both the encoding and decoding blocks. This approach supersedes traditional deep learning techniques such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs).

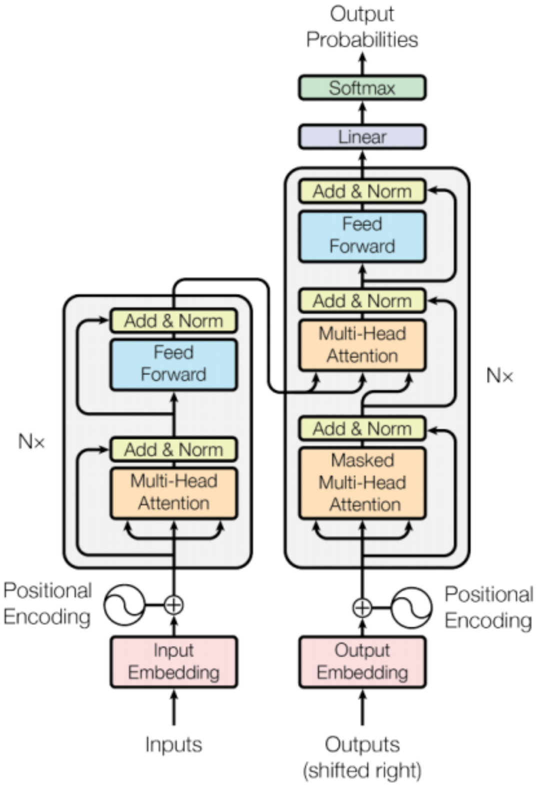

Figure 1: The Transformer-model architecture[21].

One of the significant advantages of the Transformer model is its ability to process all text in parallel during training without requiring any loops. Additionally, the attention mechanism inside the Transformer model can connect distant words to the current word, which greatly enhances the efficiency of text training.

The Transformer model is a sequence-to-sequence network that is entirely based on the attention mechanism and replaces recursive and convolutional operations. It consists of two main components: the encoder and decoder. The encoder contains two subnets: the multi-head attention mechanism and the feedforward network, while the decoder contains an additional labeled multi-head attention mechanism that only outputs results at certain points. However, the internal encoder of the Transformer model does not include convolutional or recursive loop networks, making it unable to handle text timing and resulting in the attention mechanism's calculation results remaining unchanged regardless of word position changes. To address this limitation, a position encoding mechanism has been introduced in the multi-head attention Transformer model[22]. The position of words is encoded to obtain the position embedding of the words before inputting the text into the Transformer model, and the word embeddings of the other words are added as input to the encoder and decoder.

Pre-training models are neural networks trained on vast amounts of data and saved for later use, serving as a foundation for solving similar problems. When faced with new tasks, instead of training a new model from scratch, one can begin with the pre-trained model and perform minimal fine-tuning to adapt it to the new task. This approach has led to significant progress in natural language processing, as pre-trained models can learn general language representations from large-scale texts and then fine-tune them to specific downstream tasks. As a result, most natural language processing tasks can achieve impressive results by leveraging pre-trained models.

BERT is a transformers model that has been pre-trained on a large corpus of English data using a self-supervised learning approach[23]. This means that the model was trained solely on raw text data without any human labeling, and an automatic process was used to generate inputs and labels from the text. DistilBERT[24], on the other hand, is a smaller and faster version of BERT that was also pre-trained using a self-supervised learning approach on the same English corpus, but with the BERT base model used as a teacher. Like BERT, DistilBERT was pre-trained on raw text data with automatic labeling, but used the BERT base model as a reference to guide its learning process[25].

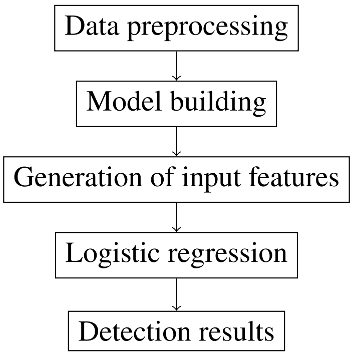

Figure 2: Architecture of proposed method

Current deep learning methods for DCC detection require extensive datasets and large amounts of time and computational resources for model training. Therefore, to achieve the sequence classification task, this project has opted to fine-tune the DistilBERT model. This involves training the model to obtain a semantic representation of text that contains rich semantic information[26], which can be utilized for DCC detection.

In addition, covert channels are communication networks that use regular channels as carriers and can bypass system security policy settings to transmit sensitive data between different security entities. However, there are differences between regular domain names and domain names with covert channels in semantic association information, such as domain name length, type, character frequency, and special resource record type. Therefore, from the perspective of natural language processing, the detection of DNS covert channels can be generalized as a binary classification task for regular or covert network domain names[27].

3.3 Loss Function

The cross-entropy loss function is a widely used approach for classification tasks, including in the DCC detection model proposed in this study. It quantifies the dissimilarity between the predicted probability distribution and the actual probability distribution. For this task, the actual probability distribution is represented as a one-hot vector indicating whether the sample belongs to the DCC or regular DNS traffic category. The predicted probability distribution is obtained by feeding the input data into the model and applying the softmax function to obtain the probability for each category.

$$L = \frac{1}{N}\sum_i L_i = \frac{1}{N}\sum_i -\left[y_i \cdot \log\left(p_i\right) + \left(1 - y_i\right) \cdot \log\left(1 - p_i\right)\right]$$

In the provided equation, y_i represents the true label of sample i, which is either 1 for DCC or 0 for regular DNS traffic. The predicted probability of sample i being a DCC traffic is represented by p_i. The loss function is then applied to all samples in the dataset, where N represents the total number of samples. The goal is to train the model to accurately predict whether a given network domain name sequence is a DCC or a regular DNS traffic by minimizing the cross-entropy loss function on the labeled dataset.

4. Experiment

4.1 Dataset

The study of DNS covert channels has been an interesting research topic in both domestic and foreign academic communities. Researchers have constructed various DNS covert channel datasets using specific tools, such as dnscat2, dns2tcp, iodine, cobalt strike, and others. In this experiment, a portion of the dataset was constructed using these tools, and the table below shows the details of the dataset.

Table 3: DCC dataset

Tools Type	Number of DCC Samples
cobalt_strike	1244
dns2tcp	5626
dnscat2_1	5172
dnsexfiltrator	13 038
dnslivery	222
iodine_1.1	259
OzymanDNS	4165
reverse_dns_shell	389
tcp-over-dns	2420
tuns	1479

A total of 73,965 samples were collected for this experiment, including 58,707 DCC samples and 15,258 regular DNS samples. The DCC samples were collected from 10 different tools, as well as some privately collected data, to ensure a diverse range of data. To ensure effective model training, the dataset was split into training, validation, and testing datasets in a 6:2:2 ratio.

Table 4: Total dataset

Sample Type	Number of Samples	Number of DCC Samples	Number of DNS Samples
DCC Dataset	34 014	34 014	0
Private Dataset	39 951	24 693	15 258
Total Dataset	73 965	58 707	15 258

4.2 Experimental Setups

In this study, various tools and technologies were utilized to conduct the experiments. Python3 and Pytorch were specifically chosen as the software frameworks to implement the proposed model due to their rich set of tools and libraries that are essential for machine learning and deep learning tasks. Moreover, the experiments were conducted on a personal computer running the latest Windows 11 operating system. To expedite the computation process, the researchers utilized an NVIDIA GeForce GTX 1060 GPU as the accelerator.

4.3 Experimental Results

In the first experiment, the proposed DCC detection method demonstrated excellent performance, with an average training loss of only 0.0406 and an average validation loss of only 0.0031. These two loss values are indicators of the model's training effectiveness. Additionally, the detection accuracy was up to 99.9%, which suggests an outstanding degree of match between the detection results and the actual results.

Table 5: Experiment 1 of the proposed method

Method	Average Training Loss	Accuracy	Average Validating Loss
Transformer	0.0406	99.9%	0.0031

The results of Experiment 1 demonstrate that the binary classification task of network domain names is highly accurate and easily manageable. The proposed method's high accuracy may be attributed to several factors, including the powerful ability of the DistilBERT model, the clear classification characteristics between regular DNS traffic and DCC traffic, and the adequate and properly prepared training datasets.

4.4 The Controlled Experiment

To further evaluate the detection performance of the model, Experiment 2 was conducted with a reduced dataset size of only 2000 samples, which accounts for only 2.7% of the original dataset. In addition, to simulate real-world scenarios, the proportion of DCC samples was reduced to only 500, while 1500 regular DNS traffic samples were selected. This controlled experiment aimed to test the few-shot learning ability of the proposed method.

Table 6: Dataset of few-shot learning

Sample Type	Number of Samples
Total Dataset	2000
Black Samples of DCC Traffic	500
White Samples of DNS Traffic	1500

Regarding the Controlled Experiment, the results show that even with a reduced dataset size and few-shot learning, the model still exhibits outstanding detection accuracy of 99.9%. However, the higher average training loss of 0.5746 and the average validating loss of 0.1063 suggest that the dataset size has an impact on the model's training performance.

Table 7: The controlled experiment of few-shot learning

Dataset	Average Training Loss	Accuracy	Average Validating Loss
Full	0.0406	99.9%	0.0031
Few-Shot	0.5746	99.9%	0.1063

4.5 The Comparison Experiment

Furthermore, to compare the detection performance of the proposed model, a comparison experiment using a simple LSTM-based detection model was conducted. The Long Short-Term Memory (LSTM) is a special type of Recurrent Neural Network (RNN) architecture designed to address the problem of vanishing and exploding gradients during long sequence training. LSTMs are known to perform better than regular RNNs in longer sequences[28].

Table 8: The comparison experiment of simple LSTM

Method	Dataset	Average Training Loss	Accuracy	Average Validating Loss
Simple LSTM	Full	0.0215	99.4%	0.021
	Few-Shot	0.3023	76.9%	0.302

The experimental results demonstrate that when using the full dataset, the accuracy of the LSTM-Based Detection method reached 99.4%, which is a good result, although slightly weaker than the proposed method. However, in the case of few-shot learning, the accuracy dropped to 76.9%, which is an unsatisfactory result.

The aforementioned experiments indicate that the binary classification of network domain names is highly compatible with deep learning methods. The DistilBERT model consistently outperforms the LSTM-based model in the DCC detection task, especially under the condition of few-shot learning. In general, the proposed method displays superior performance in DCC detection.

5. Conclusion

The successful application of the DistilBERT-based detection method in the binary classification task of network domain names for regular DNS and DCC traffic represents a significant milestone in the field of network security. By leveraging the powerful capabilities of the Transformer model in the field of artificial intelligence and deep learning, along with the covert channel detection method for DNS network protocols, this project has achieved efficient and excellent detection results, even with a limited number of samples for training. The proposed method has the potential to contribute to the development of effective and robust solutions for detecting DCC traffic in real-world network environments.

6. Future Work

However, there is still room for improvement in the future. One important direction for future work is to deploy the proposed model in real network environments to validate its effectiveness and practicality. This would entail testing the model's performance under various network conditions and configurations, as well as evaluating its ability to detect different types of covert channels in DNS protocols.

Another important aspect is to further improve the model's detection accuracy, especially for more complex and advanced covert channels. This could be accomplished by expanding the training dataset, exploring more sophisticated training techniques, and potentially integrating other machine learning or deep learning models into the detection process.

Moreover, network security systems demand real-time monitoring and detection capabilities. Therefore, it is crucial to enhance the model's lightweight design and reduce its computational and memory requirements. This could be achieved by optimizing the model architecture, enhancing the feature selection process, and exploring new hardware acceleration techniques such as GPU-based computing or edge computing.

In summary, this project has demonstrated the potential and effectiveness of integrating Transformer-based deep learning models with covert channel detection methods in network security. There are numerous promising avenues for future research and development in this area.

Acknowledgments

This work is supported in part by the Xiejialin Project of Institute of High Energy Physics under Grant no. E25467U2 and the specialized project for cybersecurity and informatization in the 14th Five-Year Plan of CAS under grant no. WX145XQ12.

References

[1] Jiarong Wang, Tian Yan, Dehai An, Zhongtian Liang, Chaoqi Guo, Hao Hu, Qi Luo, Hongtao Li, Han Wang, Shan Zeng, et al. A comprehensive security operation center based on big data analytics and threat intelligence. *Proceedings of Science*, 378:28, 2021.

[2] Tian Yan, Hao Hu, Dehai An, Fazhi Qi, and Chen Jiang. Cyber security monitoring and data analysis at ihep. *Proceedings of Science*, 351:11, 2019.

[3] Junyi Liu, Zhenyu Li, Jiarong Wang, Tian Yan, Dehai An, Caiqiu Zhou, and Gang Chen. A weakly-supervised method for encrypted malicious traffic detection. *Proceedings of Science*, 415:27, 2022.

[4] Asaf Nadler, Avi Aminov, and Asaf Shabtai. Detection of malicious and low throughput data exfiltration over the dns protocol. *Computers & Security*, 80:36–53, 2019.

[5] Yue Wang, Anmin Zhou, Shan Liao, Rongfeng Zheng, Rong Hu, and Lei Zhang. A comprehensive survey on dns tunnel detection. *Computer Networks*, 197:108322, 2021.

[6] Peng Yang, Ye Li, and Yunze Zang. Detecting dns covert channels using stacking model. *China Communications*, 17:183–194, 2020.

[7] Salvatore Saeli, Federica Bisio, Pierangelo Lombardo, and Danilo Massa. Dns covert channel detection via behavioral analysis: a machine learning approach. *arXiv:2010.01582*, 2020.

[8] Anna L Buczak, Paul A Hanke, George J Cancro, Michael K Toma, Lanier A Watkins, and Jeffrey S Chavis. Detection of tunnels in pcap data by random forests. In *Proceedings of the 11th Annual Cyber and Information Security Research Conference*, pages 1–4, 2016.

[9] Irvin Homem, Panagiotis Papapetrou, and Spyridon Dosis. Entropy-based prediction of network protocols in the forensic analysis of dns tunnels. *arXiv:1709.06363*, 2017.

[10] Irvin Homem, Panagiotis Papapetrou, and Spyridon Dosis. Information-entropy-based dns tunnel prediction. In *Advances in Digital Forensics XIV*, pages 127–140, 2018.

[11] Jawad Ahmed, Hassan Habibi Gharakheili, Qasim Raza, Craig Russell, and Vijay Sivaraman. Real-time detection of dns exfiltration and tunneling from enterprise networks. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 649–653. IEEE, 2019.

[12] Wenyue Zhang, Wei Chen, Zhen Zhang, Fan Zhang, and Lifa Wu. Itransformer_cnn: A malicious dns detection method with flexible feature extraction. *Available at SSRN 4177994*, 2022.

[13] Chang Liu, Liang Dai, Wenjing Cui, and Tao Lin. A byte-level cnn method to detect dns tunnels. In *2019 IEEE 38th International performance computing and communications conference (IPCCC)*, pages 1–8. IEEE, 2019.

[14] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324, 1998.

[15] Franco Palau, Carlos Catania, Jorge Guerra, Sebastian Garcia, and Maria Rigaki. Dns tunneling: A deep learning based lexicographical detection approach. *arXiv:2006.06122*, 2020.

[16] Alex Graves and Alex Graves. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, pages 37–45, 2012.

[17] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45:2673–2681, 1997.

[18] Jiacheng Zhang, Li Yang, Shui Yu, and Jianfeng Ma. A dns tunneling detection method based on deep learning models to prevent data exfiltration. In *Network and System Security*, pages 520–535, 2019.

[19] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern recognition*, 77:354–377, 2018.

[20] M Zhang, HL Sun, and P Yang. Identification of dns covert channel based on improved convolutional neural network. *Journal on Communications*, 41(1):169–179, 2020.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[22] Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. Ammus: A survey of transformer-based pretrained models in natural language processing. *arXiv:2108.05542*, 2021.

[23] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*, 2018.

[24] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv:1910.01108*, 2019.

[25] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *Chinese Computational Linguistics*, pages 194–206, 2019.

[26] Santiago González-Carvajal and Eduardo C Garrido-Merchán. Comparing bert against traditional machine learning text classification. *arXiv:2005.13012*, 2020.

[27] Shaomin Zheng and Meng Yang. A new method of improving bert for text classification. In *Intelligence Science and Big Data Engineering*, pages 442–452, 2019.

[28] Shaojie Chen, Bo Lang, Hongyu Liu, Duokun Li, and Chuan Gao. Dns covert channel detection method using the lstm model. *Computers & Security*, 104:102095, 2021.