Raw Packet Data Ingestion with Transformers for Malicious Activity Classifications

Nitin Sharan*, Thomas Quig,†, Eric Goodman‡, Yung Ryn Choe§, Dalton Brucker-Hahn¶
Sandia National Laboratories
Email: *nsharan@sandia.gov, †thomasquig.dev@gmail.com, ‡elgoodm@sandia.gov, §yrchoe@sandia.gov, ¶dabruck@sandia.gov

2023 International Conference on Machine Learning and Applications (ICMLA) | 979-8-3503-4534-6/23/$31.00 USG | DOI: 10.1109/ICMLA58977.2023.00330

***Abstract*—Traffic diversity, novel attacks, and sheer volume of network traffic creates a significant challenge in detecting and identifying malicious actions against a network. Due to these factors, human auditing of network events is unfeasible, requiring advanced approaches, such as machine learning techniques. Furthermore, the increasing novelty of network attacks render traditional filtering mechanisms unable to handle such threats. In this paper, we propose a novel, natural language processing approach to detecting malicious, network-based attacks, using ByT5. Our approach classifies network packet data as malicious or benign. ByT5 is a token-free sequence to sequence model, enabling the model to take in raw packet streams and classify the packets without feature extraction or preprocessing via an encoding schema. The results of our approach in classifying traffic as benign or malicious indicates promising results. Namely, when applied to the ISOT dataset, our approach achieves a maximal recall of 0.834 and a maximal F1 score of 0.693.**

***Index Terms*—seq2seq, Deep Learning, Classification, ByT5, NLP, Network Intrusion, NIDS**

I. Introduction

The continuous transition of commerce, socialization, and even infrastructure to networked systems has created an ever-increasing target for adversaries. With the goals of disruption, compromise, and sale of proprietary data, adversaries have a strong incentive to conduct malicious actions against networked systems. As noted in [8], critical infrastructure has also become an appealing target for malicious actors. Due to these factors, the arms-race of between cyber security professionals and adversaries has created a need for continuous improvement in detection capabilities against such attacks.

We posit that processing and classifying sequences of bytes from network packet data is akin to analyzing sequences of characters in natural language text; instead of processing natural language, we propose *reading* network data through state-of-art natural language processing (NLP) models. Transformer models are have been shown to be very successful in a wide range of text-based processing tasks [6], [15], [16].

For this study, we focus on the Byte T5 (ByT5) transformer model and its suitability to the task of malicious network traffic detection [17]. ByT5 is attractive for network packet classification due to its ingestion of raw bytes. The typical NLP pipeline first requires tokenization, but as network traffic has no obvious equivalent, processing raw bytes directly is a natural fit for our problem. Tokenization, within this context, refers to taking unstructured data, such as a block of text, and converting it into a set of discrete elements.

As a part of this research, we have created a framework for classifying raw packet data as malicious or benign with very little required in the way of preprocessing. This framework leverages pretrained models to reduce the burden of training the model on the end user. By using a model which does not rely on a tokenizer, we have removed a computationally costly setup stage from our framework. The contributions of this work can be summarized as follows:

- To the best of our knowledge, the first utilization of a transformer model, namely ByT5, for raw packet ingestion and malicious traffic classification.
- Thorough evaluation of ByT5 against the Information Security and Object Technology Cloud Intrusion Detection (ISOT IDS) dataset [3].
- Presentation of performance against ISOT IDS dataset with maximal recall and F1 scores of 0.834 and 0.693, respectively.

The remainder of this work is structured as follows: Section II presents the current state-of-art and relevant research relating to machine learning approaches in network traffic classification. Section III provides an overview of the dataset utilized and the necessary preprocessing steps to accomplish this work. Section IV presents a high-level overview of the ByT5 model and the training setup adopted for this work. Section V highlights the results and analysis of the trained model. Finally, section VI provides the conclusions of this work and proposes recommendations for future research directions.

II. Related Works

Previous work in the area of machine learning applications for malicious traffic classification has explored a range of techniques and models. Namely, previous efforts have examined packet processing approaches, the application of sequence to sequence (seq2seq) models for traffic classification, and efforts to label existing network traffic datasets. In particular to this work, we explore the application of utilizing a transfer learning approach to classify observed network traffic using the ByT5 model. As part of our analysis of related work, we also discuss previous efforts tangential to our own which apply natural language processing models for malicious network traffic detection.

Closely related to this research effort is previously completed work by Goodman, *et al.* [10]. In this work, the authors explore the application of Word2Vec against packet data for

DOI 10.1109/ICMLA58977.2023.00330

automatic feature extraction. From there, these features can be fed into a classification model. This approach, Packet2Vec, solves the tokenization issue by dividing the byte stream into n-grams, and then trains the embedding layer on sequences of these n-grams. Packet2Vec leverages the fact that learned embedding vectors semantically represent how the byte sequences are used within the packet data. The number of vectors in a Packet2Vec embedding layer is 2^{8n} where n is the number of bytes in the n-gram. In contrast, the model leveraged in this work, ByT5, has an embedding layer of size 256, one for each possible byte value. Packet2Vec uses a simple aggregation approach to create a fixed-sized vector for each packet, essentially a bag-of-words approach, whereas our approach utilizes a sequence of bytes, rendering better classification performance.

Similar to Packet2Vec, Mimura, *et al.* [13] apply a similar approach to vectorizing packet data, by utilizing an extension of Word2Vec, Doc2Vec, which creates a vector representation over paragraphs or documents. In this case, a single packet is treated as a document and tokens are extracted from fields in TShark summaries of the packets. In contrast, our approach reads the raw bytes of the packet with no translation or tokenization. Additionally, by distilling the packet into a single vector with Doc2Vec, there is a loss of ordering information found in the sequence, which our approach utilizes.

In this work, we utilize the ISOT IDS dataset, which fortunately comes with packet-level labels to train a supervised machine learning classifier. However, producing labels in non-research settings is problematic. Ishibashi, *et al.* [11] propose leveraging preexisting Network Intrusion Detection Systems (NIDS) to find malicious network traffic and simultaneously label it. Individual packets that triggered NIDS alarms are labeled with the corresponding alert. This work can potentially be used as bootstrap mechanism in the wild for supervised approaches such as our own. Of course, what can be learned is limited to what can already be caught by existing NIDS, but may have more generalizability than simple rule-based approaches.

Comparable to our approach, previous research has studied the use of seq2seq models for classification tasks. Work by Boutell, *et al.* propose solutions to the problem of classification when class labels are not mutually exclusive [4]. While the classification of packets as malicious or benign is a mutually exclusive scenario (packets are involved in either malicious or benign activity), the data which composes the packet can be a mix of "malicious-adjacent" and "benign-adjacent". In this way, the classification applied to packets can be seen as existing on a spectrum. You, *et al.* explore a tangential domain to network traffic classification where they apply the seq2seq model approach on Vehicle Defect information for the purpose of classifying vehicle complaint data [18]. The VDIF-M model proposed in this work uses a Bidirectional Long Short-Term Memory (LSTM) as the encoder.

TABLE I: ISOT Dataset Overview

	Dec 8th	**Dec 9th**	**Dec 15th**	**Dec 16th**	**Dec 19th**
Benign	2,095,393	3,109,678	7,148,529	1,354,095	1,579,939
Malicious	1,429	3,179,931	4,405,564	720,862	905,316
Total	2,096,822	6,289,609	11,555,093	2,074,957	2,485,255
Total (ISOT)	2,099,524	6,293,326	11,562,532	2,076,752	2,487,853
Total (pcap)	2,100,075	6,294,365	11,566,288	2,077,662	2,489,748

The ISOT dataset is a publicly available dataset which contains network traffic from a real-world cloud environment. This table provides a breakdown on the amounts of packets which were benign and malicious for each day in the ISOT dataset's first phase.

A number of previous works have examined the application of machine learning techniques for the purposes of network intrusion detection. Dong, *et. al* employ AE-Alexnet, a variation of the popular computer-vision model which utilizes auto-encoding techniques, on the task of intrusion detection [7]. In this work, the authors demonstrate that the use of machine learning for network traffic and intrusion classification is a viable avenue of research within this domain. Similarly, Chen, *et. al* explore a larger range of machine learning models, from support vector machines to deep belief networks and convolutional neural networks [5]. The outcomes of this work suggest that the application of models previously trained for unrelated or tangential tasks to traffic classification is a promising direction of research. This work expands upon this idea by using ByT5, a transformer model, to train a network traffic classifier for the purpose of malicious network traffic detection.

When it comes to the application of nature language processing models to the task of network traffic classification, there is work similar in nature to what is achieved in this paper. Lin, *et al.* propose a NIDS, referred to as PELAT, based around the BERT transformer model [12]. However, their approach only addresses unencrypted HTTP traffic, and only utilizes text-like key-value pairs from HTTP headers such as User-Agent, Content-Length, etc., and does not appear to handle raw packet data as does our approach.

Additional considerations regarding the use of machine learning approaches within NIDS, is the resource requirements for the model and the ability for the model to operate in real-time. Ngo, *et. al* consider a hardware-based approach by implementing a trained model in a Field Programmable Gate Array (FPGA) [14]. Implementation of a model in a FPGA allows for the speed and efficiency benefits of dedicated hardware without overhead and spinoff cost of fabricating discrete hardware. An extension of this work could be to apply our proposed model within such an environment and explore the benefits of hardware-assisted machine learning for malicious network classification.

Overall, while great strides have been made in constructing more sophisticated NIDS via the use of deep learning, the lateral use of NLP models for the task of network intrusion detection is a relatively untouched field. We seek to address the issue of preprocessing and spin-up time requires in creating a network traffic classification model through our use of a token-less model, ByT5.

III. Dataset Overview and Preprocessing

To evaluate our proposed approach, we make use of the Information Security and Object Technology (ISOT) Cloud Intrusion Detection System (IDS) dataset. Further, we refine this dataset using preprocessing techniques before ingesting the dataset into our model for training and testing purposes. The details of the ISOT dataset and the specific preprocessing steps taken are included below.

A. ISOT Dataset

The ISOT Research Lab is an organization whose primary directive is the rigorous development of secure computing systems and the protection of these systems. Additionally, they make publicly available a collection of datasets for use in analysis and machine learning model creation. One such dataset, the Cloud Intrusion Detection System (IDS) dataset, is leveraged as part of this work due to its applicability to the domain studied. Consisting of 2.5 terabytes of network traffic, system logs, performance data (e.g. CPU, RAM utilization), along with observed system calls [3] The Cloud IDS dataset was collected in real-world cloud environments and contains traffic from human-generated sources and automated processes. The dataset has two phases: 5 days of traffic from December 2016 and 6 days in February 2018. For our approach and model training, we initially train upon the data collected in the first time period, December 2016. Then, to evaluate the trained model, we test against the data collected in the second time period, February 2018. Table I provides an overview and details regarding the size and content of the dataset, such as the proportion of observed and labeled malicious activity relative to benign activity. During this investigation, we discovered a discrepancy between the total number of packets with labels, the totals reported by the dataset maintainers, and the totals found within the packet capture files. Our investigation indicates that this discrepancy is likely due to a lack of labeling information for a small percentage of packets. We highlight this discrepancy in Table I.

B. Preprocessing

During initial exploration of the dataset, it was discovered that preprocessing techniques would need to be applied to the data before being utilized within our proposed model. Firstly, the network traffic data lacked label information that indicated whether a packet corresponded with malicious traffic or benign traffic. Furthermore, the data has identifying information, such as IP addresses, MAC addresses, and ports which would need to be abstracted before used in model training. Particularly, this identifying information can lead to the model targeting false patterns for classification, such as labeling traffic from particular IP addresses as malicious rather than identifying the content of the traffic as benign or malicious. Scapy [9], a popular packet manipulation tool was utilized to filter packets for training. Additionally, each individual packet was assigned a label for the purposes of model training. The IDS dataset contained packet capture files with the raw packet data and comma-separated values files with timestamps, labels, and other identifying information for each packet. The packet labels could be matched to their corresponding packet capture file by correlating the packet label with a matching timestamp. We note that while this process is susceptible to collisions during packet matching, the collisions that did occur had matching labels on all affected packets, resulting in no labeling discrepancies.

TABLE II: Model Training Overview

Day	Training Instances	Training Size (GB)	Time per Epoch (hrs)	Est. Time 50 Epochs (days)
8th, 9th	6,362,720	25	28.5	59.4
15th	8,811,128	34	37	77.1
16th	1,441,724	5.6	6	12.5
19th	1,810,632	7	8	16.7

This table provides size of the data for each day from the first phase of the ISOT dataset along with the time the ByT5 model took to train one epoch on that given day

IV. ByT5 Model and Training Setup

ByT5 is a transformer model based upon another transformer model called mT5 [19], a multilingual version of T5 [17]. The novel component of ByT5 is the fact that it does not require a tokenizer; it instead encodes each possible byte value as a token for a total of 256 tokens.

A. ByT5 Model

ByT5 comes in few different ”sizes”, referring the number of parameters in the model. For the scope of the current experiments, the team used the small variant with 300 million parameters. We utilize the pretrained weights available from the Hugging Face website [1]. The model was pretrained on the mC4 dataset [2]; we leave as future work the evaluation of pretraining the model on a self-supervised learning task based upon raw packet data of our target domain. This paper focuses on the efforts of fine tuning the model to make it applicable for the task of classifying packets as malicious and benign.

B. Training Setup

As with any supervised machine learning problem, it is important to test the generalizability of the trained model by carefully partitioning the data into a training set and unseen test data. Our methodology for partitioning the data is by day. Phase 1 contains data from five days, January 8th, 9th, 15th, 16th, and 19th, 2016. We train on one-days' worth of data, and then evaluate on the other days. However, we combine the 8th and 9th days as the 8th has very few malicious examples; the ISOT data creators intend for the 8th to be useful as normal traffic in anomaly-detection approaches. We do not believe a cross-validation approach over the entire Phase 1 is appropriate, as it will likely lead to overly optimistic results; many individual attacks will have packet examples in both training and validation sets.

Next, the partitioned data is equalized, such that the ratio of benign packets matched the malicious packets. Finally, the packet data in each partition is truncated into the first 100,

TABLE III: ISOT Dataset Model Performance

Training Day	Packet Max Length	F1	Precision	Recall	Test Day
8th, 9th	100	0.602	0.574	0.632	15th
8th, 9th	200	0.111	0.449	0.063	15th
8th, 9th	400	0.101	0.463	0.0568	15th
8th, 9th	100	0.511	0.36	0.871	16th
8th, 9th	200	0.0027	0.541	0.00133	16th
8th, 9th	400	0.089	0.422	0.0499	16th
8th, 9th	100	0.496	0.36	0.784	19th
8th, 9th	200	0.130	0.066	0.007	19th
8th, 9th	400	0.06	0.318	0.036	19th
15th	100	0.585	0.434	0.90	8th, 9th
15th	400	0.483	0.464	0.504	8th, 9th
15th	100	0.550	0.433	0.754	16th
15th	400	0.56	0.52	0.612	16th
15th	100	0.693	0.592	0.834	19th
15th	400	0.53	0.63	0.461	19th

This table provides a breakdown on how each model (differentiated by maximum packet length and the day it was trained on) performed against a given test day.

200, and 400 bytes of the packets. Each of these partitions were then used to train a model which is evaluated on the other partitions. Training results with an equalized dataset were found to be better than without equalizing.

As an encoder-decoder model, ByT5 requires a predicted output sequence. To satisfy its need for an output sequence, the model is adapted to predict the string ”malicious” and ”benign”.

The systems used to train and test the models featured either an Intel Xeon Gold 6240R or Xeon Silver 4214 as their processors. For their GPUs, these systems used Nvidia’s Tesla V100 (32Gb) or the Quadro RTX 8000.

V. Results and Analysis

A. Results

The initial goal was to have 50 epochs of training, but training ran for a couple of weeks; each individual epoch took somewhere between six to 37 hours to complete depending on the size of packets given to the model and the size of the partition it is trained on. Table II lists the training resources needed for the 100-byte models.

B. Analysis

As can be seen in Table III, preliminary testing found that the model trained on the 15th tends to have the highest True Positive Rate and F1 score, likely due to the fact that the 15th had the most malicious examples. In addition, the model trained on the 16th has a low True Positive Rates but had high precision; this could be explained by the theory that attacks conducted on the 16th were limited in variety but were also conducted on the other days. Furthermore, when comparing the 100-byte models to their 200- and 400-byte counterparts, there seems to be no discernible improvement in results when truncating less bytes. Below is a table of the most performant epochs.

Looking at the results in Table III, it appears that 100 byte models have the highest F1 and Recall scores. Considering the high volume of traffic in a network, false positives should be minimized to avoid unnecessary work or clutter for a network administrator. Since false positives are to be discouraged, a high precision score is an important benchmark.

C. Convergence

Due to the complexity of the model and level of resources required for fine-tuning it, finding the optimal number of epochs to run is paramount. Testing the model epoch by epoch can yield useful data in determining whether or not a given model has converged yet.

Looking at the testing data in Figure 1, subfigures 1a & 1b, it seems that, for the 8th & 9th 100-byte model, further training will not improve the model’s performance; other steps must be taken to improve the model’s performance.

The 15th 100 byte model falls into the same category as the 8th & 9th model, not requiring any further training. In regard to the longer byte length models, the greater complexity of the data suggests that more epochs will be necessary to get the models to a similar competency to the 100-byte max length models.

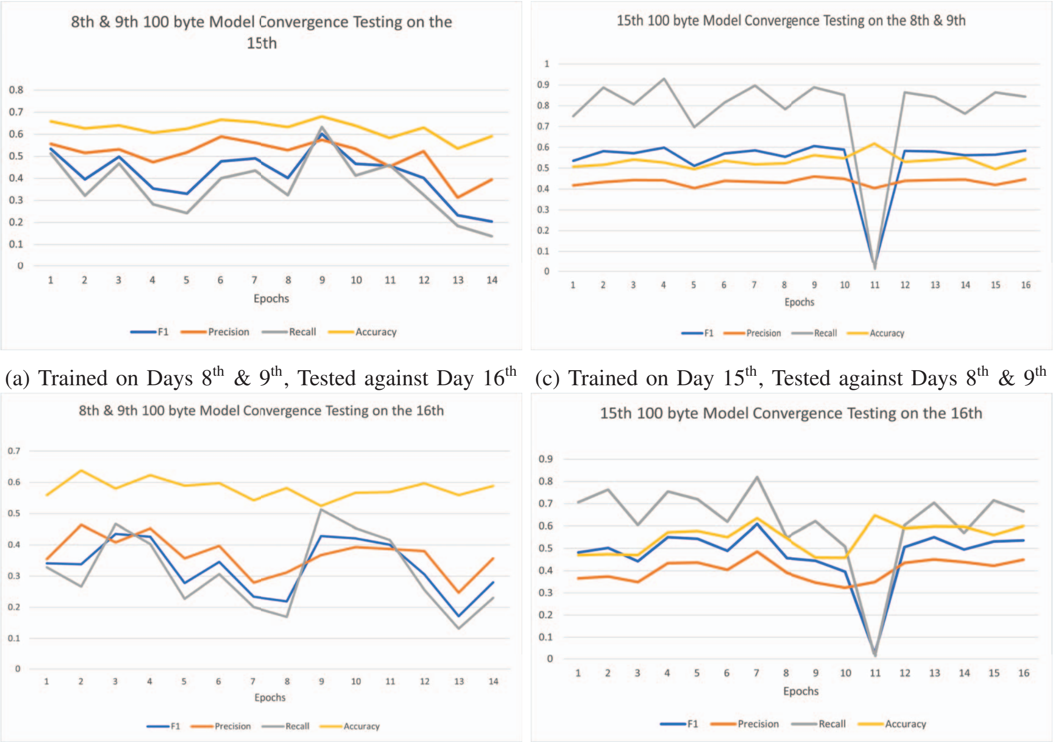
VI. Future Work and Conclusions

Future work for this model would focus on multiclass classification of packet data as opposed to the binary classification that is currently done. As it stands, the sequence to sequence output on the encoder side is a poor fit for this style of problem. Future improvements could involve integrating additional data sources to improve the model’s accuracy or to detect other malicious vectors. These additional data sources include system calls, memory dumps, and OS logs. These data sources would have to be lined up/correlated with the packet data. This process may be facilitated by comparing time stamps.

Virtualization allows us to enhance the capabilities of the model by providing it host data via monitoring of the host’s statistics via a containerized daemon. Docker runtime metrics can be collected through the docker stats command; this data is rich in information and can be transformed into a variety of file types, including JSON, for human readability and easy ingestion by the model.

Integration with a host for an online version of the model would involve running a local daemon to collect data at intervals to be preprocessed and given to an instance of the model. This program would work as follows.

When combining host and network analytics, there are two primary design choices. The first choice is an ensemble system which would combine host and network data by using multiple classifiers suited for each specific category of data. Next, a model stacking approach would be used with one final model aggregating the output of the multiple classifiers. The second choice is to use a single integrated model. For a single model, the network and host data would be combined first and then



(a) Trained on Days 8th & 9th, Tested against Day 16th (c) Trained on Day 15th, Tested against Days 8th & 9th

(b) Trained on Days 8th & 9th, Tested against Day 16th (d) Trained on Day 15th, Tested against Day 16th

Fig. 1: Convergence Graphs for 100 Byte Models: under the current configuration and parameters, our models show convergence. We do note that there were issues with vanishing gradients, but evidence of convergence is still strong in our proposed approach.

```
Data: None
Result: Normalized and Truncated Host Traffic Data
while TRUE do
    results := get_packet_traffic();
    normalized_results := normalized(results);
    truncated_data := truncate(normalized_results);
    send_data_to_model(truncated_data);
    sleep(some period);
end
```

vectorized. These vectors would be fed into the model for malicious activity classification.

The preliminary survey work done here is rather promising. The training speed and evaluation speed of the model, especially when truncating packet data to 100 bytes is quick enough for online evaluation of malicious traffic given traffic rates presented by the ISOT dataset. Further work and fine-tuning would have to be done to improve the accuracy of the model and to prevent high rates of false positives. One area of note is the hand-holding that may be required when training higher complexity models. Care must be taken to make sure that the model avoids the vanishing/exploding gradient issue. Using equalized datasets mitigates this problem, but there's still a chance it may occur.

Acknowledgment

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This article describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the article do not necessarily represent the views of the U.S. Department of Energy or the United States Government. This work was supported through contract #70RSAT21KPM000105 with the U.S. Department of Homeland Security Science and Technology Directorate.

References

[1] Hugging face. huggingface.co. Accessed: 2023-06-07.

[2] mc4 dataset. tensorflow.org/datasets/catalog/c4#c4multilingual. Accessed: 2023-06007.

[3] Abdulaziz Aldribi, Issa Traoré, Belaid Moa, and Onyekachi Nwamuo. Hypervisor-based cloud intrusion detection through online multivariate statistical change tracking. *Computers & Security*, 88:101646, 2020.

[4] Matthew R. Boutell, Jiebo Luo, Xipeng Shen, and Christopher M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.

[5] Lin Chen, Xiaoyun Kuang, Aidong Xu, Siliang Suo, and Yiwei Yang. A novel network intrusion detection system based on cnn. In *2020 Eighth International Conference on Advanced Cloud and Big Data (CBD)*, pages 243–247, 2020.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[7] Yuansheng Dong, Rong Wang, and Juan He. Real-time network intrusion detection system based on deep learning. In *2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS)*, pages 1–4, 2019.

[8] Chandelis Duster. Energy secretary says adversaries have capability of shutting down us power grid — cnn politics, Jun 2021.

[9] N. Gift and J.M. Jones. *Python for Unix and Linux System Administration*. O'Reilly Series. O'Reilly, 2008.

[10] Eric L. Goodman, Chase Zimmerman, and Corey Hudson. Packet2vec: Utilizing word2vec for feature extraction in packet data, 2020.

[11] Ryosuke Ishibashi, Kohei Miyamoto, Chansu Han, Tao Ban, Takeshi Takahashi, and Jun'ichi Takeuchi. Generating labeled training datasets towards unified network intrusion detection systems. *IEEE Access*, 10:53972–53986, 2022.

[12] Ling-Hsuan Lin and Shun-Wen Hsiao. Attack tactic identification by transfer learning of language model, 2022.

[13] Mamoru Mimura and Hidema Tanaka. Reading network packets as a natural language for intrusion detection. In Howon Kim and Dong-Chan Kim, editors, *Information Security and Cryptology – ICISC 2017*, pages 339–350, Cham, 2018. Springer International Publishing.

[14] Duc-Minh Ngo, Andriy Temko, Colin C. Murphy, and Emanuel Popovici. Fpga hardware acceleration framework for anomaly-based intrusion detection system in iot. In *2021 31st International Conference on Field-Programmable Logic and Applications (FPL)*, pages 69–75, 2021.

[15] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[16] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020.

[17] Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. Byt5: Towards a token-free future with pre-trained byte-to-byte models, 2022.

[18] Xindong You, Yuwen Zhang, Baoan Li, Xueqiang Lv, and Junmei Han. Vdif-m: Multi-label classification of vehicle defect information collection based on seq2seq model. In Yuyu Yin, Ying Li, Honghao Gao, and Jilin Zhang, editors, *Mobile Computing, Applications, and Services*, pages 96–111, Cham, 2019. Springer International Publishing.