

Intrusion Detection System for AI Box Based on Machine Learning

Jiann-Liang Chen, Zheng-Zhun Chen, Youg-Sheng Chang,
Ching-Iang Li, Tien-I Kao, Yu-Ting Lin, Yu-Yi Xiao, Jian-Fu Qiu
*Department of Electrical Engineering, NTUST (National Taiwan University of
Science and Technology), Taipei, Taiwan*
ITRI (Industrial Technology Research Institute), Hsinchu, Taiwan

Abstract—This paper presents an integrated application of network intrusion detection. The intercepted packets are first analyzed using a machine learning algorithm, the HistGradient Boosting classifier, to detect network traffic as abnormal or normal. If the network traffic is unnatural or disruptive, the system will immediately notify the information security expert whether to disrupt the network traffic. We propose an information security application combined with hardware and software. The application system can secure IoT devices or any field that needs to use the network environment, including abnormal network traffic, system outages, betrayals, and other cyber threats. The system is scalable and can easily port to other devices or platforms. This study will introduce datasets and pre-processing methods, machine learning model building, abnormal packet handling, and operating system environment building of the AI BOX device. Two public datasets were used to train and test the model, and our model obtained 99.9% prediction accuracy. The system has been successfully tested on the Yocto Project operating system of the AI BOX device.

Keywords—Artificial Intelligence, Information Security, Machine Learning, IoT Device Security, HistGradient Boosting Classifier Algorithm, Confidentiality of Data Transfer, Packet Capture Analysis, Feature Selection.

I. INTRODUCTION

With the successful development of Industry 4.0, various highly automated industries such as semiconductors and smart factories need to operate in the form of the Internet of Things (IoT) due to various needs, so there are often accompanying cyber security incidents. Related incidents have been commonly seen in Taiwan enterprises, such as the TSMC ransomware incident in 2018 [1]. Taiwan, which is often the target of attacks, has also started to focus on information security protection in the industrial field in recent years. Traditional information security event handling is shown in Figure 1.

Reference [2, 3] mentioned that weaknesses in network devices are often one of the reasons for attacks on Industry 4.0 environments, including distributed denial of service (DDoS), Bonet, and other types of attacks by sending many anomalous packets. A large number of devices in smart factories transmit

data over the network, and some of them even use cloud services. Therefore, network device defense is the next trend in information security applications in the industry.

In this paper, we combine CICFlowMeter and machine learning algorithms with ITRI's AI BOX to propose a new automated intrusion detection system device that uses artificial intelligence to identify malicious traffic and is easily portable to act as a router or gateway between external networks and industrial networks to create an integrated information security solution.

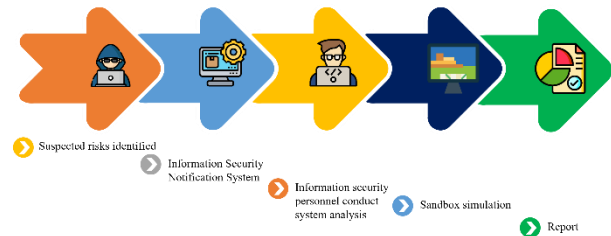


Figure 1. Traditional information security risk process flow

II. RELATED WORK

Choi, H et al. [4] proposed an intrusion detection model based on AutoEncoder and tested different types of autoencoder with the highest accuracy of 91.70%. However, this study was only tested on the NSL-KDD dataset, collected in 2009 and too old to represent the current network attacks. The actual utility of this model would be reduced due to this reason.

Islam, F.B et al. [5] proposed an Artificial Neural Network (ANN)-based model and evaluated the accuracy of the model in the experimental stage using the Mean Absolute Percentage Error (MAPE) equation and found that the accuracy of the model increases as the MAPE value decreases. The model was tested using a publicly available evaluation dataset, and the highest accuracy of nearly 97% was obtained.

Alzahrani, A.O et al. [6] proposed an intrusion detection system for SDN controllers, first analyzing and normalizing the data. This study selected five features for model training and tested three baseline algorithms, such as decision tree, XGBoost,

and Random Forest. The experimental results showed that XGBoost achieves the highest accuracy rate and performs well in all attack detection.

Y. Dong et al. [7] collected data in real time based on the Flume and Flink engine, cleaned the data, and extracted and trained the features with the proposed Auto-Encoder AlexNet model to obtain 94.32% accuracy. However, this study might have problems determining whether there was an intrusion threat on time due to the long training time and lack of portability.

L. Zhiqiang et al. [8] used deep learning algorithms and the complete UBSW-NB15 dataset, which introduced cross-validation techniques and tested the effect of 10-fold, as well as the Gedeon method, to identify the effect of essential features on model error. Finally, six machine learning methods, both supervised and unsupervised, were tested, and this study proposed method obtained the highest accuracy of 99.5%.

III. PROPOSED APPROACH

This section details the approach proposed in this study, including the network traffic acquisition technique, the machine learning algorithm selected, the dataset for model training and its processing, and the environment configuration and algorithm concatenation.

A. HistGradient Boosting Classifier Algorithm [9]

This study uses a machine learning model, the HistGradient Boosting Classifier algorithm, inspired by the LightGBM [10] algorithm and based on the Gradient Boosting Classifier algorithm [9].

The HistGradient Boosting Classifier algorithm is a tree structure. Instead of using Gini coefficients as in the Random Forest algorithm [11], the training data is first classified into several numerical segments, i.e., the training data is histogrammed, and the splitting points are determined based on the structure of the data after that.

Since the HistGradient Boosting Classifier algorithm uses a histogram approach, there is no need to sort the values each time a feature is selected for splitting, compared to the Gradient Boosting Classifier algorithm. The HistGradient Boosting Classifier algorithm can speed up several sets when datasets exceed 10,000.

B. Dataset

The Intrusion Detection System (IDS) has been one of the tools that have long helped information security personnel prevent cyber attacks. For machine learning, the training dataset is often one of the essential parts, so suitable datasets for the industrial network environment need to be selected for training. This study uses the following three datasets for model training with the Random Forest algorithm. Intrusion detection evaluation dataset, DDoS evaluation dataset, and our own collected SSH-Brute Force dataset.

1) CIC-IDS 2017 (Intrusion Detection Evaluation Dataset)

The CIC-IDS 2017 dataset [12] collected real-world data, including normal and abnormal network traffic through the CICFlowMeter network traffic analysis tool, and conducted various common network attacks in five days,

which is very suitable for training binary classification or multiclass machine learning models. It was also mentioned in [13] that this data set contains some shortcomings, such as a Huge Volume of Data, Missing Values, High-class imbalance, etc., so how to perform data preprocessing is also a focus of this research.

2) CIC-DDoS 2019 (DDoS Evaluation Dataset)

The CIC-DDoS 2019 [14] dataset evaluated various publicly available DDoS attack datasets in previous years, but there was no dataset for reflection-based DDoS attacks; unlike the previous dataset, the CIC-DDoS 2019 dataset added various standard network devices in the experimental field to match the real-world usage environment, and conducted eleven types of DDoS attacks, and classified them into reflection-based and exploitation-based attack techniques, and finally presented 80 features from the data.

3) SSH-Brute Force dataset

Figure 2 is the flow chart of how we collect the brute force attack, in which we use the brute force cracking tool named Hydra [15] in Kali Linux, a brute force tool for cracking account passwords. It uses the existing account password files to compare and crack each combination. After nearly half a day of data collection, through CICFlowMeter to extract the features and add them to the training dataset.

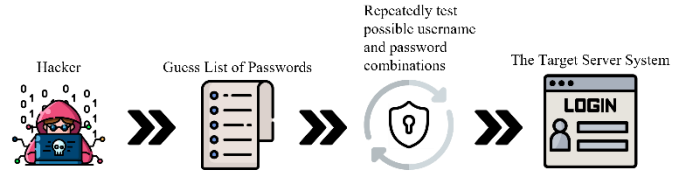


Figure 2. Flowchart for collecting SSH-Brute Force attacks dataset

C. Data Process

In this study, three different datasets are used, two public and one collected by this study.

In the dataset processing, the three datasets were mixed and split into training and test sets with a ratio of 3:1. Then, the datasets were further analyzed for label uniformity and found to have unbalanced labels. In this study, we use the imblearn [16] tool to perform data under-sampling to equalize the labeling ratio of the whole dataset. We then use the feature importance evaluation tool `f_classif` [9] and the feature selection tool `SelectKBest` [9] to perform feature filtering. First, the `f_classif` tool uses ANOVA (Analysis of Variance) statistics to assign P-values to each feature. Then the `SelectKBest` tool ranks the P-values of each feature and selects K features in the dataset for model training. Twenty features are finally selected for model training in this study. Table 1 below shows the names of the selected features.

Table 1. Selected dataset features

Flow duration	Bwd pkt len max	Bwd pkt len min	Bwd pkt len mean
Bwd pkt len std	Flow iat std	Flow iat max	Fwd iat tot

Fwd iat std	Fwd iat max	Pkt len min	Pkt len max
Pkt len mean	Pkt len std	Pkt len var	Pkt size avg
Bwd seg size avg	Idle mean	Idle max	Idle min

After selecting the desired dataset features, data pre-processing is performed. The data pre-processing techniques use in this study are StandardScaler [9], MinMaxScaler [9], and LabelEncoder [9]. First, StandardScaler is applied to each feature value to make the Gaussian distribution of each feature value so that the mean value of each group feature is zero and the standard deviation is one. Then MinMaxScaler is performed to reduce the range of each feature in the dataset from zero to one, and finally, the labeled data in the dataset is encoded by LabelEncoder.

D. CICFlowMeter [17, 18]

In this study, we use CICFlowMeter, a network traffic feature extraction tool, which is a Pcap file and obtains more than 80 features after analysis and extraction. The features include packet statistics such as maximum length, average length, and standard deviation. The CICFlowMeter tool can be used to obtain network traffic features for subsequent analysis.

E. AI BOX Environment

The AI BOX embedded system uses the Yocto Project [19], which provides a flexible set of tools, software stacks, and configurations to create custom Linux images for embedded and IoT devices or where a custom Linux operating system is required. The release version is called Poky. Figure 3 below shows the development architecture of the Yocto Project.

The build environment is the output directory of the build release, which contains the two .conf files automatically generated when building the environment. The four options listed in Metadata Layers can all have their paths added to bblayers.conf file. Build output is the build output files, such as reference images, licenses, SDKs, etc.). The OpenEmbedded build system is the base file for building the distribution; the Poky file is downloaded to your computer, providing a base building environment. It provides a basic environment for building a Linux development version.

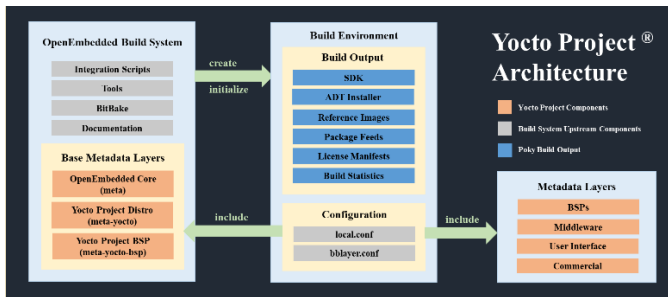


Figure 3. Yocto Project architecture

Yocto's workflow is shown in Figure 4 below. The first part starts with the developer specifying an architecture, policies, patches, and configuration details. The second part is building the system. Fetches and downloads the source code wherever set. The project supports standard methods such as source code repositories systems. After downloading, the source files are

extracted to the local workspace from where the patch is applied, and the configuration and compilation software will run in normal steps. In the third part, the software is installed into a temporary staging area where choices binary package format is used to roll up the software. We will be checking different QA and sanity in the entire build process. Finally, there is the fourth part. After the binary has been created, a root file image will be generated to produce a file system image.

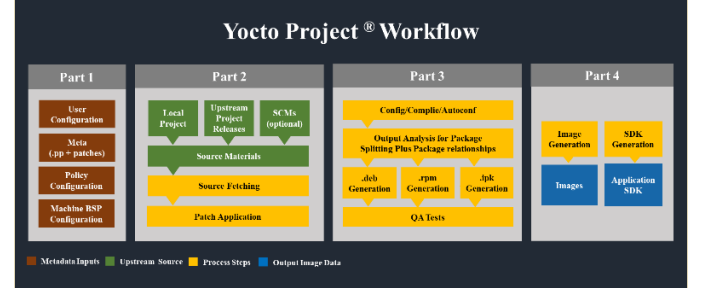


Figure 4. Yocto Project workflow

Cross-Development Toolchain Generation is used to build mirrors and applications on target hardware. Figure 5 shows the advanced build environment for toolchain construction and use. When running BitBake to create an image, the OpenEmbedded build system uses the host's gcc compiler to bootstrap the file called gcc-cross. The compiler is the tool BitBake uses to compile the source files. The Build Host produces three toolchains:

- 1) *gcc-cross*: builds the target images.
- 2) *gcc-crosssdk*: a transitory toolchain that produces relocatable code that executes on the SDKMACHINE.
- 3) *gcc-cross-canadian*: which executes on the SDKMACHINE, produces and runs to create target applications.

And the Target Device runs the Target Images and Target Applications.

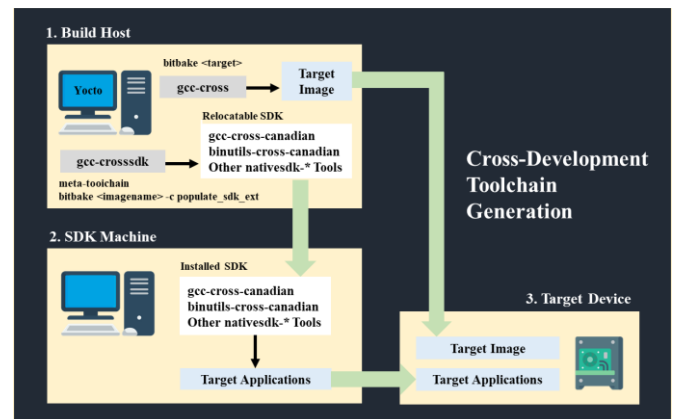


Figure 5. Cross-Development toolchain generation

F. PROCEDURE ARCHITECTURE

Figure 6 shows the structure of our model training framework. First, we balance the data by UnderSampling to balance the overall number of abnormal and normal labels, then

calculate the p-value through ANOVA, select 20 features into StandardScaler, MinMaxScaler, and LabelEncoder to normalize and encode the data, and finally classify the data into normal and abnormal through HisGradient Boosting Classifier.

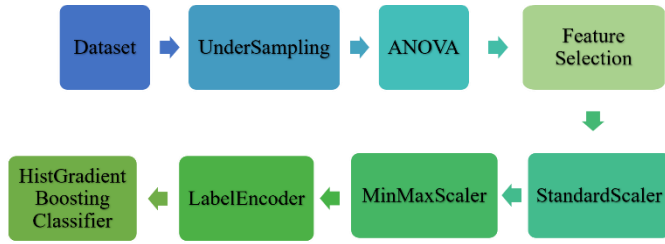


Figure 6. Data processing flow chart

Figure 7 shows the application of AI BOX and IDS in the industrial field. The abnormal traffic will be filtered by AI BOX. When an intrusion attack occurs, abnormal packets cause the system to generate an intrusion alert, notifying information security professionals for follow-up and research.

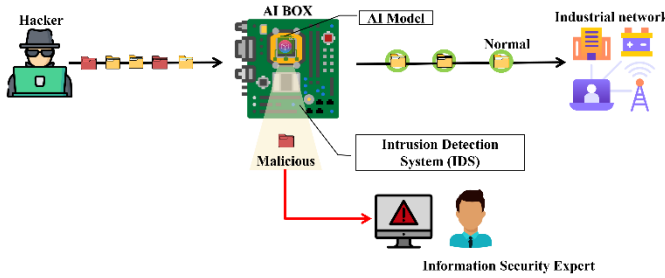


Figure 7. The role of the AI BOX in the industrial environment

IV. RESULT

This study uses a machine learning algorithm to train a model with three datasets and combine them with CICFlowMeter, a network traffic feature acquisition tool, to build an autonomous network traffic detection system. The algorithm design can be divided into two parts: autonomous network traffic acquisition and network traffic prediction. First, when the system is executed, network traffic will be captured continuously and archived in order. Second, the network traffic prediction part will independently predict network traffic files and generate relevant prediction reports for each batch of network traffic files. Generate alarms when abnormal network traffic is detected, and move their related information to the abnormal network traffic temporary storage area for subsequent information security personnel to view.

In machine learning model training, various machine learning models are tested, and their results are analyzed. Figure 8 below shows the histograms of the performance evaluation metrics of different machine learning algorithms.

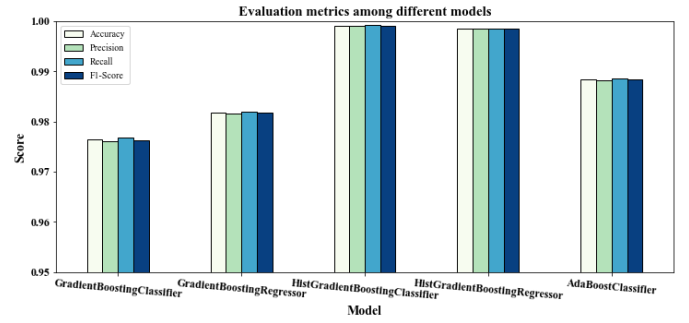


Figure 8. The performance evaluation metrics of different machine learning algorithms

We mainly try regression and classification models in the machine learning algorithm section. These include gradient-based regression and classification models: Gradient Boosting Regressor, Gradient Boosting Classifier, Regression and classification models that use histogramming of data to speed up without losing accuracy: HistGradient Boosting Regressor, HistGradient Boosting Classifier. AdaBoost Regressor is a regression model that uses decision tree regression with weight adjustment according to the prediction error.

The above model evaluation found that the classification model performs better than the regression model for large data sets. The overall execution efficiency and prediction accuracy of the histogram-optimized model are excellent, and the HistGradient Boosting Classifier algorithm is used for network traffic prediction in this study. Figure 9 shows the confusion matrix of the model on the test dataset with all four model evaluation metrics above 0.999.

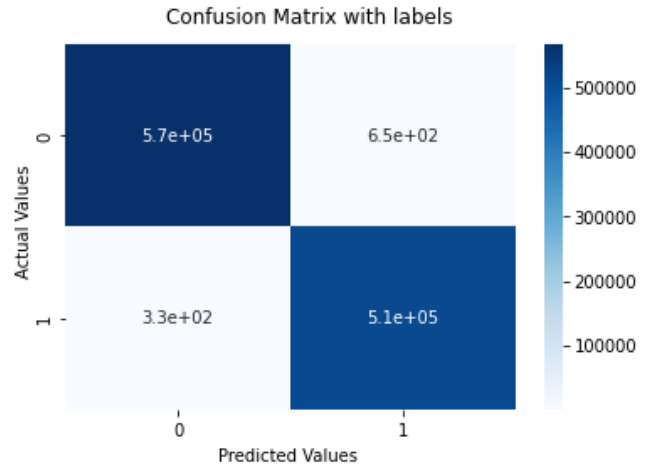


Figure 9. The confusion matrix of the testing dataset

The flow monitoring algorithm design and model training is completed. This study combines the process architecture with the AI BOX module proposed by ITRI to realize a complete autonomous abnormal network traffic detection mechanism. Figures 10 and 11 below show the normal and abnormal network traffic reports predicted at different times, respectively. The abnormal network traffic will be alerted, and the network traffic file will be moved to the temporary storage area.

```

cicflowmeter stopped
////flow predict////
cicflowmeter started
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
-----
Time           : Mon Oct 24 16:28:10 2022
Amount of flows : 24
Percentage of Anomaly : 0.0
Percentage of Legit : 100.0
-----

```

Figure 10. The normal network traffic report

```

cicflowmeter stopped
////flow predict////
cicflowmeter started
-----
Time           : Mon Oct 24 16:28:30 2022
////////Alert//////// : Anomaly traffic detected
flow file      : flow/flows3.csv
-----

```

Figure 11. The abnormal network traffic report

A. Model Evaluation Metrics

The following four metrics are based on the confusion matrix.

- 1) *True Positive (TP): the label of the data is abnormal, and the predicted result is abnormal*
- 2) *True Negative (TN): the label of the data is normal, and the predicted result is normal*
- 3) *False Positive (FP): the label of the data is normal, and the predicted result is abnormal*
- 4) *False Negative (FN): the label of the data is abnormal, and the predicted result is normal*

Based on TP, TN, FP, and FN metrics, we derive the following metrics:

- Accuracy:

$$\frac{TP + TN}{(TP + TN + FP + FN)} \quad (1)$$

- Precision:

$$\frac{TP}{(TP + FP)} \quad (2)$$

- Recall:

$$\frac{TP}{(TP + FN)} \quad (3)$$

- F1-score:

$$\frac{2 \times Precision \times Recall}{Precision + Recall} \quad (4)$$

V. CONCLUSION

In this study, we propose an autonomous intrusion detection system based on machine learning in conjunction with ITRI AI

BOX. We first use the CICFlowMeter tool to capture network traffic packets and then extract network traffic features as input for the machine learning model to perform prediction. This model is in the test dataset and achieves almost 99.9% accuracy and 99.9% recall. With AI BOX and Autonomous Intrusion Detection System, smart factories or sites using network transmission can be protected from abnormal intrusion attacks. In addition, we built our embedded system environment with the Yocto Project, and the lightweight advantage of Yocto enables AI BOX to be more scalable. Until now, there has been no research on the information security of IoT devices based on the Yocto Project.

ACKNOWLEDGMENT

We thank the ITRI (Industrial Technology Research Institute), Taiwan, for supporting the “Customized Subsystem Anomaly Packet Detection Information Security Module Project” and providing the embedded system device AI BOX.

REFERENCES

- [1] (2018) CommonWealth Magazine [Online]. Available: <https://english.cw.com.tw/article/article.action?id=2194>.
- [2] V. Pedreira, D. Barros and P. Pinto, "A Review of Attacks, Vulnerabilities, and Defenses in Industry 4.0 with New Challenges on Data Sovereignty Ahead," *Sensors*, vol.21, no. 15, pp.5189, 2021.
- [3] P. Zorić, M. Musa and T. M. Kuljanić, "Smart Factory Environment: Review of Security Threats and Risks," Perakovic, D., Knapcikova, L. (eds) *Future Access Enablers for Ubiquitous and Intelligent Infrastructures. FABULOUS 2021. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, vol.382. Springer, Cham, 2021.
- [4] H. Choi, M. Kim, G. Lee and W. Kim, "Unsupervised learning approach for network intrusion detection system using autoencoders." *The Journal of Supercomputing*, vol. 75, no. 9, pp.5597–5621, 2019.
- [5] F. B. Islam, R. Akter, D. S. Kim and J. M. Lee, "Deep Learning Based Network Intrusion Detection for Industrial Internet of Things," *Proceedings of the Korea Telecommunications Society Conference*, pp.418-421, 2020.
- [6] A.O. Alzahrani and M.J.F. Alenazi, "Designing a Network Intrusion Detection System Based on Machine Learning for Software Defined Networks," *Future Internet*, vol.13, no.5, pp.111, 2021.
- [7] Y. Dong, R. Wang and J. He, "Real-Time Network Intrusion Detection System Based on Deep Learning," *Proceedings of the International Conference on Software Engineering and Service Science*, pp.1-4, 2019.
- [8] L. Zhiqiang, G. Mohi-Ud-Din, L. Bing, L. Jianchao, Z. Ye and L. Zhijun, "Modeling Network Intrusion Detection System Using Feed-Forward Neural Network Using UNSW-NB15 Dataset," *Proceedings of the International Conference on Smart Energy Grid Engineering*, pp.299-303, 2019.
- [9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol.12, pp.2825-2830, 2011.
- [10] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Y. Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," *Proceedings of the*

- International Conference on Neural Information Processing Systems*, pp.3149-3157, 2017.
- [11] L. Breiman, "Random Forests," *Machine Learning*, vol.45, pp.5–32, 2001.
 - [12] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," *Proceedings of the International Conference on Information Systems Security and Privacy*, pp.108-116, 2018.
 - [13] R. Panigrahi and S. Borah, "A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems," *International Journal of Engineering & Technology*, vol.7. no. 3.24, pp.479-482, 2018.
 - [14] CIC-DDoS 2019, <https://www.unb.ca/cic/datasets/ddos-2019.html>, last accessed 2022/9/21.
 - [15] van Hauser Heuse, M. (2021). hydra (Version 9.2) [Computer software]. <https://github.com/vanhauser-thc/thc-hydra>
 - [16] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: a python toolbox to tackle the curse of imbalanced datasets in machine learning," *The Journal of Machine Learning Research*, vol. 18, pp.559–563, 2017.
 - [17] A. H. Lashkari, G. D. Gil, M. S. I. Mamun and A. A. Ghorbani, "Characterization of Tor Network traffic using Time based Features," *Proceedings of the International Conference on Information Systems Security and Privacy*, pp.253-262, 2017.
 - [18] G. D. Gil, A. H. Lashkari, M. S. I. Mamun and A. A. Ghorbani, "Characterization of Encrypted and VPN Network traffic using Time-related Features," *Proceedings of the International Conference on Information Systems Security and Privacy*, pp. 407-414, 2016.
 - [19] Yocto Project Homepage, <https://www.yoctoproject.org/>, last accessed 2022/10/1.