Analysis of NetFlow Features' Importance in Malicious Network Traffic Detection

Adrián Campazas-Vega[1(✉)], Ignacio Samuel Crespo-Martínez[1], Ángel Manuel Guerrero-Higueras[1], Claudia Álvarez-Aparicio[1], and Vicente Matellán[2]

[1] Robotics Group, University of León, Campus de Vegazana S/N, 24071 León, Spain
{acamv,icrem,am.guerrero,calvaa}@unileon.es
[2] Supercomputación Castilla y León (SCAyLE), Campus de Vegazana S/N, 24071 León, Spain
vicente.matellan@scayle.es
http://robotica.unileon.es

Abstract. Malicious traffic detection allows for preventing cybersecurity-related threats. Machine learning algorithms are commonly used to detect such traffic in computer networks by analyzing packets. In wide-area networks, such as RedCAYLE (Red de Ciencia y Tecnología de Castilla y León), it is not possible to analyze every packet routed. So we pose that in such networks sampled flow data may be used to provide malicious traffic detection. This work presents the analysis carried out of the relevance that every NetFlow feature has in the K-Nearest Neighbors (KNN) algorithm in order to detect malicious traffic. Validation of the model has been carried out with real network data from RedCAYLE. Results show that it is necessary to train the models with sampled flow data. They also show that the *nexthop* feature has a negative influence on malicious traffic detection in wide-area networks such as RedCAYLE.

Keywords: Netflow features analysis · K-Nearest Neighbors (KNN) · Network traffic · Machine learning · Network security · Malicious traffic detection

1 Introduction

Cybercriminals have been professionalized using different techniques to achieve profit through the theft of victims' data. They use a large number of techniques, such as port scanning, malware distribution, lateral network movements,

The research described in this article has been partially funded by Instituto Nacional de Ciberseguridad de España (INCIBE), under the grant "ADENDA 4: detección de nuevas amenazas y patrones desconocidos (red Regional de Ciencia y Tecnología)", addendum to the framework agreement INCIBE-Universidad de León, 2019–2021; the Spanish Ministry of Science, Innovation, and Universities RTI2018-100683-B-I00 grant; and the regional Government of Castilla y León under the grant BDNS (487971).

J. J. Gude Prego et al. (Eds.): CISIS 2021/ICEUTE 2021, AISC 1400, pp. 52–61, 2022.
https://doi.org/10.1007/978-3-030-87872-6_6

privilege escalation, zero-day attacks, and social engineering. Most attacks generate malicious traffic. Such traffic may be used to detect and stop them. In the literature, various techniques have been used to detect malicious traffic. For instance, in [18], the authors present a payload-based anomaly detector for intrusion detection. In [19], the authors characterize the benign network traffic so that anomalous traffic patterns may be detected. Different works show that it is possible to detect malicious traffic using machine learning models. In [16] authors propose an anomaly detection system using supervised learning algorithms that have previously been trained with the UNSW-NB15 dataset. The authors obtained 94.36% accuracy using the AODE algorithm, 92.70% using Bayesian networks, and 75.73% using the NB algorithm. This work was validated by the same authors in [17], in which, using the same dataset, they tried to verify which algorithms were the best in detecting malicious traffic in terms of accuracy and efficiency. The results showed that the AODE algorithm obtained the best results with an accuracy of 97.26% and a running time of approximately 7 s.

In order to detect malicious traffic, in [23] a hybrid model is proposed which implements bagging and tree rotation techniques. To test the efficiency of the generated models, the authors used the NSL-KDD and UNSW-NB15 datasets, obtaining a precision of 85.8%. Finally, in [22] the authors carried out a review of the literature in which they collected what are the machine learning algorithms and datasets most used to detect malicious traffic. Although it is possible to detect malicious traffic using machine learning models, all the above proposals used network packets as input for the different algorithms. In wide-area networks, it is not possible to analyze every packet that the network routes. That is why this type of infrastructure uses flow-based technologies.

A flow is defined as a set of packets that pass through an observation point in the network during a specific time interval. All packets belonging to a particular flow have a common set of properties, such as source and destination IP addresses, and source and destination port numbers [6,8]. Flow data do not store the payload of network packets, reducing the computational power required to process flows versus complete packages. In addition to those already mentioned, some of the features that flow data gathers are the number of packets that the flow contains, the IP protocol, the TCP flags, and the timestamp. One of the most widely used flow-based technologies is NetFlow. NetFlow was designed by Cisco as a network protocol for collecting network statistics [5]. NetFlow is deployed on most commercial routers.

However, the network traffic volume that some routers manage is so large that even using NetFlow it is not possible to use every packet to generate flow data. So they need to sample the packets before generating flows. For instance, RedCAYLE, which manages traffic belonging to universities and research centers in Castilla y León, manages an average amount of traffic of 6.7 Gb/s. To decrease the computational load of gathering flow data on the routers, a sampling threshold is applied. The sampling threshold used by RedCAYLE routers is 1 packet out of 1000.

This work comes from the results obtained in [4]. In this work, the authors developed a framework for collecting flow datasets using NetFlow. With such datasets, the authors fitted supervised learning models, obtaining that the best classifier to detect port scans was KNN.

This work presents an analysis of NetFlow's features to determine which are most relevant when training the KNN algorithm. The training has been carried out with two different datasets, the first one gathering sampled flow data and the second one gathering unsampled flow data. Validation has been carried out against real data obtained from RedCAYLE in order to double-cleck if the fitted model is able to detect attacks in a real environment and how the features used to fit it influence the detection of malicious traffic.

The remainder of the paper is organized as follows: Sect. 2, describes the materials and tools used in this job, as well as the methodology used to evaluate the importance of NetFlow features to detect malicious traffic; Sect. 3 shows the results obtained in the experiment and discusses the results. Finally, the conclusions are presented in the Sect. 4.

2 Materials and Methods

Section enumerates the materials used, presents the experiments carried out, and describes the methods used to evaluate the results obtained in the experiments.

2.1 NetFlow

NetFlow [7] is a protocol developed by Cisco Systems to collect information about the network traffic. The usage of NetFlow has become so popular in recent years that manufacturers such as Juniper and Enterasys Switches support this technology. NetFlow was introduced as a new feature of Cisco's routers to provide IP traffic gathering allowing administrators to have a global vision of what is happening in the infrastructure they manage.

NetFlow has multiple versions: NetFlow V1,V5 and V9. RedCAYLE uses NetFlow V5. For such release, the features that flow data gather are enumerated in Table 1.

2.2 RedCAYLE

RedIRIS [21], is a Spanish research network that provides communications services to the scientific and university community. RedCAYLE manages RedIRIS network traffic belonging to the region of Castilla y León. It is managed from SCAYLE. Network traffic management is carried out through two main routers located in León and Valladolid, and a series of auxiliary switches and routers distributed throughout the region.

RedCAYLE provides to its affiliated institutions (educational centers, universities, hospitals, scientific facilities, etc.) a high-capacity communications backbone infrastructure, which allows for accessing both the resources of the research

Table 1. NetFlow features

Feature	Description
sysuptime	Current time in milliseconds since the export device started
unix_secs	Current count of seconds since 0000 UTC 1970
unix_nsecs	Residual nanoseconds since 0000 UTC 1970
engine_type	Flow switching motor type
engine_id	Slot number switching engine flow
exaddr	Flow exporter Ip
srcaddr	Source IP address
dstaddr	Destination IP address
nexthop	IP address of the next hop router
input	SNMP index of the input interface
output	SNMP index of the exit interface
dpkts	Number of packets contained in the flow
doctets	Total number of bytes of layer 3 in the packets of the flow
first	Sysuptime at start of flow
last	Sysuptime when the the last packet in the flow was received
srcport	TCP/UDP source port number
dstport	TCP/UDP destination port number
tcp_flags	TCP flags
prot	IP type of protocol (Por ejemplo, TCP = 6; UDP = 17)
tos	IP type of service (ToS)
src_as	Autonomous system number of the source, either source or pair
dst_as	Autonomous system number of the destination, either source or pair
src_mask	Source address prefix mask bits
dst_mask	Destination address prefix mask bits

network and the Internet. Currently, RedCAYLE offers a wide variety of services: 1 Gbps point-to-point transport service, internet connection, IP addressing, incident management, and service supervision. The RedCAYLE monitoring service allows its affiliated institutions for analyzing and diagnosing the status of their network services. The usage of NetFlow provides a network analysis based on statistics. As mentioned above, to decrease the computational load of gathering flow data, a sampling threshold of 1 packet out of 1000 is applied.

2.3 MoEv

MoEv [10] is an open-source tool developed in Python [20] for building machine learning models from datasets. The latest release accepts as input both CSV files and images. Some of the MoEv features are data cleaning, normalization, dimen-

sionality reduction, hyperparameter tuning (using the Grid-SearchCV method), and parallel running (using DASK). It evaluates several machine learning algorithms by considering accuracy, recall, precision, and F1 score. MoEv also allows for carrying out validation tests to ensure the optimal generalization of its models. Once a model, or a set of models, has been fitted, it is validated by using a different dataset.

MoEv has been used successfully in different research areas, such as in the detection of jamming and spoofing attacks in real-time location systems [11], or the prediction of academic success in educational institutions [9,12,13]. Furthermore, in [4] the authors successfully replicated the experiment proposed by [1], thus demonstrating the validity of MoEv in generating machine learning models for the detection of network attacks through flows.

2.4 DOROTHEA

DOROTHEA is a Docker-based solution that allows for creating virtual network topologies to generate and collect flow data. It uses a NetFlow sensor that builds flows from the packets that pass through a network interface. DOROTHEA is configurable and scalable, allowing, for example, to select the number of attack and victim nodes.

The framework offers two testbeds. The first one allows for simulating the generation of benign traffic. In order to get realistic responses, it is connected to the Internet. The second testbed allows for carrying out distributed network attacks in an isolated Internet environment. Isolation ensures that all the collected traffic corresponds to attack flows, thus avoiding noise when generating datasets. Testbed separation allows for the empirical labeling of the network traffic.

In both testbeds, all traffic goes through a router that incorporates a sensor that generates the NetFlow's flows from the packets it routes *ipt_NetFlow* [14]. The sensor processes network traffic and provides flow data in NetFlow V5, V9, and IPF formats. Flow data are sent to a warehouse every 2 min.

In addition to packet routing and flow data generation, the router performs packet sampling. As with commercial routers, DOROTHEA allows you to select the packet sampling rate to be used when generating flows. This feature allows for simulating the behavior of real routers, allowing researchers to generate datasets under the same conditions as big networks deployed in production. Finally, DOROTHEA returns a CSV file with the generated network flows.

2.5 Data Gathering

To carry out the experiments, two datasets have been created using DOROTHEA. Both datasets contain benign traffic and malicious traffic. Specifically, malicious traffic comes from port scanning attacks. The first dataset (aka $\mathcal{D}_1$) has been collected without packet sampling. The second dataset (aka $\mathcal{D}_2$) has been collected with a sampling threshold of 1 packet out of 1000, simulating the conditions in RedCAYLE routers.

Both datasets contain approximately 50% benign traffic. Benign flow data were labeled '0'. Malicious flow data generated were labeled '1'. Both datasets are available online under an open-access license with the following DOIs: 10.5281/zenodo.4106730 and 10.5281/zenodo.4600638.

The benign traffic was generated using three scripts developed in Python. These scripts are also available online in [3]. The first script simulates web browsing. It carries out queries in different search engines generating HTTP and HTTPS traffic. The second script simulates the sending of emails using the SMTP protocol. Finally, the third script simulates SSH connections.

Malicious traffic has been generated using the Nmap tool. Different types of port scans have been launched on both TCP and UDP ports. The scans performed were: TCP SYN; TCP Connect; UDP; TCP NULL, FIN, and Xmas; TCP ACK; TCP Window; and TCP Maimon scanning [15]. The attacks were executed by 100 machines distributed over the 65,536 ports of the 200 attacked nodes. The Python script used to carry out the attacks is available online at [2].

The range of network addresses in the training datasets was 182.168.1.1/24 for the benign traffic simulators and attack nodes, and 126.52.30.0/24 for victim nodes.

To validate the previously fitted models, a third dataset has been generated (aka $\mathcal{D}_3$), with flow data collected in the RedCAYLE.

Table 2 shows the volume of the datasets used in this work.

Table 2. Number of Flows

Dataset	#Benign flows	# Malicious flows	#Total flows
$\mathcal{D}_1$	1.429.038	1.178.992	2.608.030
$\mathcal{D}_2$	1447	1447	2894
$\mathcal{D}_3$	460	460	920

2.6 Evaluation

To fit the KNN models that have been used in the experiments, the tool MoEv and the datasets $\mathcal{D}_1$ and $\mathcal{D}_2$ have been used. Then, an analysis of the features of each model has been carried out in order to know which NetFlow features have the highest importance. In order to do so, the technique permutation feature importance has been used. This procedure breaks the relationship between the feature and the target, thus the drop in the model score is indicative of how much the model depends on the feature, obtaining, as a result, the weight that each one has in the training of the KNN model.

Once the study was carried out, the detection rate of the model was verified with the $\mathcal{D}_3$ dataset. In order to know how the features affect the detection of malicious traffic in a production environment, the models have been re-fitted and obtained their accuracy removing the most relevant features according to the previous study. The accuracy of the model has been calculated as shown in

the Eq. 1, where TP is true-positive rate, TN is the true-negative rate, FP is the false-positive rate and FN is the false-negative rate.

$$Accuracy = \frac{T_P + T_N}{T_P + F_P + T_N + F_N} \tag{1}$$

3 Results and Discussion

Table 3 shows the importance of NetFlow features, described in Table 1. The features that are not shown in the new table with respect to the older one, have an importance of 0 for the model.

Table 3. Weight for the features of the model KNN for $\mathcal{D}_1$ and $\mathcal{D}_2$

Feature	Weight $\mathcal{D}_1$	Weight $\mathcal{D}_2$
nexthop	0.266652	0.400000
srcaddr	0.196473	0.116930
dstaddr	0.160707	0.073476
last	0.107621	0.005079
first	0.107598	0.005079
sysuptime	0.107593	0.004515
unix_secs	0	0.004402
unix_nsecs	0	0.000451

As can be seen in the Table 3 the three most relevant features are nexthop, srcaddr, and dstaddr for both models, regardless of the dataset on which they have been trained. In the case of the trained model without sampling, dataset $\mathcal{D}_1$, the last, first and sysuptime features also take relevance. In the case of the model trained with sampling, dataset $\mathcal{D}_2$, in addition to the previous features, unix_secs and unix_nsecs also take relevance. The difference that we can observe in the weight of the features is that for the model trained with the dataset $\mathcal{D}_2$ the nexthop has greater weight than in the model trained with the dataset $\mathcal{D}_1$ which distributes the weight of the features more evenly between nexthop, srcaddr, dstaddr, last, first, and sysuptime.

In Table 4 we can see the accuracy of the model KNN validated with the dataset $\mathcal{D}_3$ and that has previously been trained with the datasets $\mathcal{D}_1$ and $\mathcal{D}_2$.

As can be seen in Table 4 the model trained without sampling ($\mathcal{D}_1$), cannot detect any malicious flow regardless of the features with which it has been trained. In all cases, the model detects benign traffic, but it does not detect any flow classified as attack, so the accuracy of the model is 50% in all cases. From the analysis of these data, we can concluded that the KNN model trained with $\mathcal{D}_1$ is not functional in detecting malicious network traffic in networks that perform packet-level sampling.

Table 4. True Negative, True Positive and Accuracy of KNN classifier trained with $\mathcal{D}_1$ and $\mathcal{D}_2$ based on features removed and validated with $\mathcal{D}_3$

Features removed	T_N $\mathcal{D}_1$	T_N $\mathcal{D}_2$	T_P $\mathcal{D}_1$	T_P $\mathcal{D}_2$	Accuracy $\mathcal{D}_1$	Accuracy $\mathcal{D}_2$
nexthop	100%	96.52%	0%	100%	50%	98.26%
srcaddr	100%	91.09%	0%	0%	50%	45.54%
dstaddr	100%	100%	0%	0%	50%	50%
nexthop, srcaddr	100%	91.09%	0%	0%	50%	45.54%
nexthop, dstaddr	100%	100%	0%	0%	50%	50%
srcaddr, dstaddr	100%	0%	0%	100%	50%	50%
nexthop, srcaddr, dstaddr	100%	0%	0%	100%	50%	50%

On the contrary, we can see that if we eliminate the nexthop feature, the model trained $\mathcal{D}_2$ obtains an accuracy of 98.26% when we validated it with the dataset $\mathcal{D}_3$. The rest of the combinations of features made when training the model, show an accuracy of less than 50 % discarding its validity.

From the analysis of the Table 4 we can affirm, on the one hand, that it is necessary to train models with sampled traffic to be able to detect traffic on networks such as RedCAYLE. On the other hand, the use of the nexthop feature affects negatively of the detection of malicious traffic in this type of network, therefore it is necessary to eliminate this feature when training the model. In addition, it is necessary to maintain both the source IP address and the destination IP address when training the model to increase its accuracy.

If we observe the nexthop field of the $\mathcal{D}_1$ and $\mathcal{D}_2$ datasets, we can see that in the flows classified as benign, the nexthop is always the router's laboratory that will route the packet to the Internet, however if we observe the malicious traffic flows, we can see that the nexthop field is 0, since due to the architecture of the attack laboratory, it does not have an Internet connection and therefore the jump from the attacking machine to the victim is direct.

On the other hand, if we look at the $\mathcal{D}_3$ dataset collected in RedCAYLE, the nexthop field has a much greater variance than the datasets generated with DOROTHEA since they correspond to real IP addresses and more complex network architectures. That is why it is necessary to remove this field when training models for them to work in a real environment.

4 Conclusions

The detection of malicious traffic is a good indicator in the detection of various threats. It has been shown in the literature that malicious traffic can be detected using machine learning algorithms. The difficulty in detecting these types of threats increases when it is not possible to analyze all the traffic on a network. This occurs in networks that manage a large amount of traffic per second, making it impossible to analyze all the packets that it route. To know the state of the network, this type of infrastructure uses flow-based protocols such as Netflow. Networks as large as RedCAYLE do not have enough computing power to analyze all packets, even using NetFlow. For this reason, these

networks perform packet sampling before generating flows, in this specific case RedCAYLE selects 1 packet out of every 1000.

In this work, three different datasets have been generated, $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{D}_3$. The first two datasets have been generated using DOROTHEA. Both contain approximately 50% of benign traffic and 50% of port-scanning attack traffic. The difference between both datasets is that $\mathcal{D}_2$ has been generated with a sampling threshold. Finally, $\mathcal{D}_3$ has been generated with flow data collected from RedCAYLE.

With the $\mathcal{D}_1$ and $\mathcal{D}_2$ datasets the KNN model has been fitted in order to demonstrate if it is possible to detect real malicious traffic coming from RedCAYLE. The KNN model has been selected because it is the model that has shown the best results in detecting malicious network traffic in the literature [4,22].

Once the models were trained, an analysis of the importance of NetFlow features in the generated models was carried out. This analysis reported that the features nexthop, srcaddr, and dstaddr are the most relevant when training the KNN model. With the information obtained in the analysis, the models have been re-trained eliminating the most relevant features. Once the models have been re-fitted, they have been validated with the flows obtained from RedCAYLE ($\mathcal{D}_3$).

From the validation with the $\mathcal{D}_3$ dataset, it has been shown that the results of the models trained with $\mathcal{D}_1$ do not exceed 50% accuracy regardless of the features with which are train the model. On the other hand, the model trained with the $\mathcal{D}_2$ dataset provides an accuracy of 98.26% if the nexthop feature is eliminated when training the model.

Two clear conclusions can be drawn from the experiments carried out. On the one hand, it is necessary to train models with sampled traffic to be able to detect traffic on networks such as RedCAYLE. On the other hand, the nexthop feature negatively affects the detection of malicious traffic in wide-area networks, and it is necessary to remove this feature in the training phase of the model.

References

1. Boukhamla, A., Coronel, J.: CICIDS 2017 dataset: performance improvements and validation as a robust intrusion detection system testbed. Int. J. Inform. Comput. Secur. **9** (2018)
2. Campazas-Vega, A., Crespo-Martínez, I.: Source code DOROTHEA attacks generation. https://niebla.unileon.es/cybersecurity/dorothea/-/tree/master/labs/lab_attacks/attacks. Accessed 13 Mar 2021
3. Campazas-Vega, A., Crespo-Martínez, I.: Source code DOROTHEA normal traffic generation. https://niebla.unileon.es/cybersecurity/dorothea/-/tree/master/labs/lab_normal/generator/generate-traffic. Accessed 13 Mar 2021
4. Campazas-Vega, A., Crespo-Martínez, I.S., Guerrero-Higueras, Á.M., Fernández-Llamas, C.: Flow-data gathering using netflow sensors for fitting malicious-traffic detection models. Sensors **20**(24), 7294 (2020)
5. Cisco: About Cisco (2021). https://www.cisco.com/. Accessed 13 Mar 2021

6. Claise, B., Zander, S.: Network working group J. Quittek Request for Comments: 3917 nec europe ltd. category: Informational t. zseby fraunhofer fokus (2004)
7. Claise, B., Sadasivan, G., Valluri, V., Djernaes, M.: Cisco systems netflow services export version 9 (2004)
8. Claise, B., Trammell, B., Aitken, P.: Specification of the IP flow information export (ipfix) protocol for the exchange of flow information. RFC 7011 (Internet Standard), Internet Engineering Task Force, pp. 2070–1721 (2013)
9. Fernández, A.G., et al.: Evaluación del resultado académico de los estudiantes a partir del análisis del uso de los sistemas de control de versiones. RIED. Rev. Iberoamericana Educación a Dist. **23**(2), 127–145 (2020)
10. Guerrero-Higueras, Á.M., Campazas-Vega, A., Crespo-Martínez, I.S.: Module evaluator (moev). Technical report, Robotics group, Universidad de León (2020). https://doi.org/10.5281/zenodo.4114127
11. Guerrero-Higueras, Á.M., DeCastro-García, N., Matellán, V.: Detection of cyber-attacks to indoor real time localization systems for autonomous robots. Robot. Auton. Syst. **99**, 75–83 (2018)
12. Guerrero-Higueras, Á.M., DeCastro-García, N., Rodriguez-Lera, F.J., Matellán, V., Conde, M.Á.: Predicting academic success through students' interaction with version control systems. Open Compu. Sci. **9**(1), 243–251 (2019)
13. Guerrero-Higueras, Á.M., Fernández Llamas, C., Sánchez González, L., Gutierrez Fernández, A., Esteban Costales, G., González, M.Á.C.: Academic success assessment through version control systems. Appl. Sci. **10**(4), 1492 (2020)
14. ipt_NetFlow: Source code ipt_NetFlow. https://github.com/aabc/ipt-NetFlow. Accessed 13 Mar 2021
15. Lyon, G.F.: Nmap network scanning: the official Nmap project guide to network discovery and security scanning. Insecure (2009)
16. Nawir, M., Amir, A., Lynn, O.B., Yaakob, N., Ahmad, R.B.: Performances of machine learning algorithms for binary classification of network anomaly detection system. In: Journal of Physics: Conference Series. vol. 1018, p. 012015 (2018)
17. Nawir, M., Amir, A., Yaakob, N., Lynn, O.B.: Effective and efficient network anomaly detection system using machine learning algorithm. Bull. Electr. Eng. Inform. **8**(1), 46–51 (2019)
18. Parekh, J.J., Wang, K., Stolfo, S.J.: Privacy-preserving payload-based correlation for accurate malicious traffic detection. In: Proceedings of the 2006 SIGCOMM Workshop on Large-Scale Attack Defense, pp. 99–106 (2006)
19. Pena, E.H.M., Barbon, S., Rodrigues, J.J.P.C., Proença, M.L.: Anomaly detection using digital signature of network segment with adaptive ARIMA model and paraconsistent logic. In: 2014 IEEE Symposium on Computers and Communications (ISCC), pp. 1–6 (2014). https://doi.org/10.1109/ISCC.2014.6912503
20. Python: About python (2021). https://www.python.org/about/. Accessed 13 Mar 2021
21. RedIRIRS: About redirirs (2021). https://www.rediris.es/rediris/index.html.es. Accessed 13 Mar 2021
22. Sobrín-Hidalgo, D., Campazas Vega, A., Guerrero Higueras, Á.M., Rodríguez Lera, F.J., Fernández-Llamas, C.: Systematic mapping of detection techniques for advanced persistent threats. In: Herrero, Á., Cambra, C., Urda, D., Sedano, J., Quintián, H., Corchado, E. (eds.) CISIS 2019. AISC, vol. 1267, pp. 426–435. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-57805-3_40
23. Tama, B.A., Comuzzi, M., Rhee, K.H.: TSE-IDS: a two-stage classifier ensemble for intelligent anomaly-based intrusion detection system. IEEE Access **7**, 94497–94507 (2019). https://doi.org/10.1109/ACCESS.2019.2928048