VHS-22 – A Very Heterogeneous Set of Network Traffic Data for Threat Detection

Paweł Szumełda
pawel.szumelda@gmail.com
Warsaw University of Technology
Warsaw, Poland

Natan Orzechowski
natan.orze@gmail.com
Warsaw University of Technology
Warsaw, Poland

Mariusz Rawski
mariusz.rawski@pw.edu.pl
Warsaw University of Technology
Warsaw, Poland

Artur Janicki*
artur.janicki@pw.edu.pl
Warsaw University of Technology
Warsaw, Poland

ABSTRACT

Researching new methods of detecting network threats, e.g., malware-related, requires large and diverse sets of data. In recent years, a variety of network traffic datasets have been proposed, which have been intensively used by the research community. However, most of them are quite homogeneous, which means that detecting threats using these data became relatively easy, allowing for detection accuracy close to 100%. Therefore, they are not a challenge anymore. As a remedy, in this article we propose a VHS-22 dataset – a Very Heterogeneous Set of network traffic data. We prepared it using a software network probe and a set of existing datasets. We describe the process of dataset creation, as well as its basic statistics. We also present initial experiments on attack detection, which yielded lower results than for other datasets. We claim that the data in the VHS-22 dataset are more demanding, and therefore that our dataset can better stimulate further progress in detecting network threats.

CCS CONCEPTS

• **Security and privacy → Network security**.

KEYWORDS

network traffic dataset, malware detection, network traffic analysis, DoS attacks, botnets, machine learning

ACM Reference Format:
Paweł Szumełda, Natan Orzechowski, Mariusz Rawski, and Artur Janicki. 2022. VHS-22 – A Very Heterogeneous Set of Network Traffic Data for Threat Detection. In *Proccedings of the European Interdisciplinary Cybersecurity Conference (EICC 2022), June 15–16, 2022, Barcelona, Spain.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3528580.3532843

*Corresponding author.

EICC 2022, June 15–16, 2022, Barcelona, Spain

ACM ISBN 978-1-4503-9603-5/22/06…$15.00
https://doi.org/10.1145/3528580.3532843

1 INTRODUCTION

The importance of machine learning (ML)-aided threat detection in computer networks has grown significantly in recent years [5]. The development of new methods in this area is directly connected with the quality, credibility and contemporaneity of the datasets available for researchers. Many high-quality datasets have been published, but when the research on detection of malicious traffic was conducted, it turned out that using these datasets does not guarantee success in detecting novel threats. One of the reasons is that existing datasets usually contain quite homogeneous traffic, therefore the ML algorithms get easily overfitted to their characteristics.

In this work we propose a new flow-level dataset named VHS-22 – a Very Heterogeneous Set of traffic data. It is a dataset of flow parameters extracted using a software network probe from five different sources of network traffic – four datasets and an Internet location. We describe the process of its creation, present its basic statistics and run initial threat detection experiments. We claim that our proposed VHS-22 dataset is more demanding, and therefore that can better stimulate further progress in detecting network threats.

Our article is structured as follows. First, in Section 2 we briefly review the available datasets containing network traffic. In Section 3 we describe a software network probe which allows aggregation of packets into flows. In Section 4 we present the process of creating the VHS-22 dataset, and describe its basic statistics. In Section 5 we present the results of initial experiments performed on VHS-22. Section 6 concludes the paper.

2 EXISTING NETWORK TRAFFIC DATASETS

According to Ring et al. [12], over the past few years more than 30 network-based intrusion detection datasets have been published. They contain either real or simulated traffic, and store data usually in the form of pcap files, flow-level data, or both.

One of the datasets used in various studies is the CICIDS 2017 dataset [15]. It contains the traffic captured during five days of activity in a simulated network. Both pcap and bidirectional flow formats have been published. These datasets cover various kinds of attacks, such as botnets, (D)DoS, web application attacks, or SSH brute-force attempts. In total, 2830540 flows were collected.

Another dataset is UNSW-NB15 [11], created in the Cyber Range Lab of the Australian Center for Cyber Security. It is a hybrid

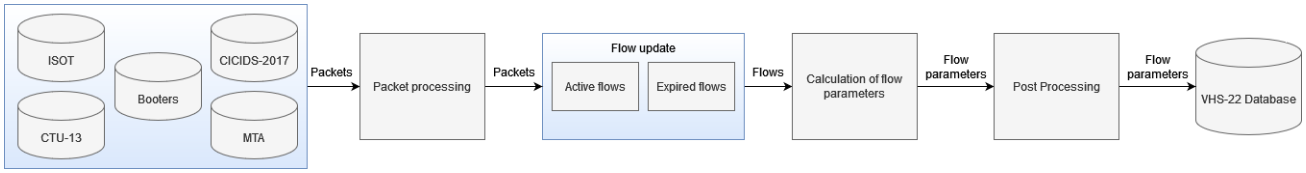

Figure 1: Process of creation of the VHS-22 dataset using software network probe

dataset that combines network attacks and real traffic originated from normal activities. 100 GB of raw traffic was captured, and the following types of the attacks were identified: fuzzing, backdoor usage, exploitation, reconnaissance, shellcode, worms, and a few more. In total, about 2.5 million records are included.

To provide better understanding of DDoS attacks offered as a service, a Booter dataset [14] has been created. This dataset contains traces of nine different booter attacks in packet-based format. More than 250 GB of network traffic have been captured. This dataset provides information such as user's geolocation, user's anonymized IPs, attack type, and attack duration.

Another dataset with botnet-type traffic is the ISOT Botnet dataset [13]. It is a blend of four separate datasets containing botnet and legitimate traffic. Botnet traffic utilizes fast flux-based DNS network and also both HTTP and P2P communication. Two datasets from the Traffic Lab at Ericsson Research in Hungary and the Lawrence Berkeley National Lab were incorporated to add everyday-use traffic from a variety of applications, including HTTP web-browsing behavior and gaming packets, as well as BitTorrent client packets.

Yet another botnet-focused dataset is CTU-13 [7], which contains flows captured in a university network. It distinguished 13 scenarios containing different botnet attacks. The dataset is available in three formats: packet, unidirectional flow and bidirectional flow. Traffic was labeled using a three-stage approach. In the first stage, all traffic to and from infected hosts is labeled as botnet. In the second stage, traffic which matches specific filters is labeled as normal, while the remaining traffic is labeled as background.

The purpose of the RoEduNet-SIMARGL2021 project [10] was to present DDoS and PortScan attacks in a real network environment. The authors also proposed classification methods and a list of 44 parameters useful in detecting those attacks. Simulation of three attacks was imposed on legitimate traffic in the RoEduNet academic network through connecting real devices to a virtualized environment representing an attacker's infrastructure.

A very useful source of malware-related network traffic data is the Malware Traffic Analysis (MTA) project [4]. It is a blog which posts pcap files and malware samples. Since 2013 more than 1800 posts have been published.

A hybrid approach to dataset creation was proposed for the KDD-MTA'19 dataset [9]. It was specifically tailored to train and evaluate ML-based malware traffic analysis algorithms. KDD-MTA'19 is a dataset merged from the Malware Capture Facility Project and the MTA repository to provide legitimate and malicious traffic, respectively. The data are periodically updated. The malicious traffic has its origin in the MTA repository only, which generates a risk of overfitting of ML algorithms.

An ideal dataset for network threat detection research should be relatively up to date, correctly labeled, and possibly contain real network traffic with various kinds of attacks. It should also encompass normal user-generated traffic and span a long period. However, most of the existing datasets are quite homogeneous. Training of detection algorithms based on such datasets makes overfitting to certain datasets tremendously likely. Moreover, network traffic generated in a lab is vastly flexible; hence it allows freedom of data manipulation. Nevertheless, such a scenario increases the risk of artifacts. On the other hand, real network traffic captured in a specific period for business or over an insufficiently long period may be underrepresented. Therefore, we claim that merging several types of datasets, containing both emulated and real world traffic, can be profitable.

3 SOFTWARE NETWORK PROBE

To create a flow-level dataset we needed to implement a software network probe which would allow us to process packets into flows. We created a probe that converts pcap files passed as an input into unidirectional network traffic flows. We decided not to use any of the existing probes (such as nfdump) to develop a fully-flexible tool for research purposes.

The traffic data is processed by the network probe by the following actions:

- process (replay) packets stored in pcap files;
- analyze packets in chosen network stack layers;
- create or update a unidirectional network stream in the active flow table;
- move inactive flows to the expired flows table;
- calculate flow parameters for the expired flows;
- store flow parameters in the output dataset.

The traffic contained in the source pcap files is replayed. The network probe loads packets from a file, analyzes them and creates a network flow or updates an existing one. Packet headers are analyzed in terms of second, third and fourth ISO/OSI Reference Model layers. Assignment of new packets to flows is based on a hash function of the header parameters.

Hash is calculated from the following parameters:

- IP source address;
- IP destination address;
- source port;
- destination port;
- transport layer protocol.

Considering the transport layer protocols, the conditions for classifying the stream as ended are RST or FIN flags in the case of TCP and reaching a predefined inactivity time in the case of UDP. The flows considered as ended are statistically analyzed and their

parameters are extracted, as described in the next section. Expired flows are dumped to a file.

The network probe was implemented in the C++ language, using the libpcap library [6]. Using the BPF packet filter, only IPv4 packets wrapped with TCP or UDP were copied to an application that works in user space.

Captured packets are processed starting with the second ISO/OSI layer. From data link layer information about the timestamp and the packet length is fetched. The *Ether_type* field contains information about the higher layer protocol used, which is, in the network probe's case, IPv4. After receiving the IP header, it is possible to decode the source and destination IP addresses, as well as the transport layer protocol. Knowing the values of the headers of transport layer protocols, it is possible to decode the recipient's port, as well as the TCP flags, if applicable.

Current flows are stored in *unordered map*, available in the C++ standard library. This map can store unsorted, unique key-value pairs. The map receives Flow class as a value, which contains all flow parameters, while FlowKey class keys are needed for computing the hash. The constructor in Flow class creates a new flow, setting parameters such as: source and destination IP address, source and destination port number, first packet timestamp, transport layer protocol.

For every incoming packet a hash of FlowKey is calculated and then checked against the existing FlowKeys in *unordered map*. If the hash does not exist, a new Flow is created, setting parameters such as: source and destination IP addresses, source and destination port numbers, first packet timestamp, transport layer protocol. If the hash already exists, the existing flow is updated. The packet count value is incremented, TCP flags are updated (if applicable), and a new timestamp and the packet size are added to the list.

In the case of the TCP protocol, the appearance of a FIN or RST flag means the end of the flow. Then, some of the flow's parameters are updated. Furthermore, the flow is moved from the active flows map to the expired flows list. Post-processing of the parameters consists in converting source and destination IP addresses to ASCII format, marking last timestamp, calculating flow's duration and total byte count, as well as their statistical parameters. They will be described in detail in the next section.

In the case of UDP packets, these are periodically checked by the application thread, which is iterating through the active flows cache. The last packet's arrival time in a flow is compared to last packet's arrival time on the network adapter and, if this exceeds the time difference by a predefined value (set in our case to 10 sec), it is moved from the current flows cache to the expired flows list.

At the end, data is moved to a text file containing data in the JSON format, so that it can be utilized, e.g., in the Apache Kafka messaging broker system.

4 CREATION OF VHS-22 DATASET

Using the network probe described in the previous section, we have created a new heterogeneous flow-level dataset named VHS-22. The creation process is described in detail in the below subsections.

Table 1: Different attacks in VHS-22 on time axis.

Attack type	Source dataset	Starting date	Length [days] (pcap files)
botnet	ISOT	2022-01-01	1
various	MTA	2022-01-02	1
webattacks	CICIDS-17	2022-01-03	1
bruteforce	CICIDS-17	2022-01-04	1
botnet	CICIDS-17	2022-01-05	1
DDoS	CICIDS-17	2022-01-06	1
DDoS	Booters	2022-01-07	5
botnet	CTU-13	2022-01-12	12

4.1 Sources of network traffic data

As the source data, we decided to combine the following datasets: ISOT, CICIDS 2017, CTU-13, Booters, and the traffic samples from MTA. This choice was justified by the following facts:

- all of these datasets and traffic samples are labeled, publicly available and open source;
- in the case of the CICIDS 2017 dataset, the network traffic was generated and collected in a simulated university network, hence its data are a reliable representation of traffic in real networks;
- the CICIDS 2017 dataset is divided into a few categories of threat including the most commonly reported attacks [8]: web attacks, web scanning, DoS and DDoS attacks, botnet traffic, brute force attacks, infiltration, and heartbleed vulnerability usage; this is a wide representation of attacks performed in real environments;
- traffic available on the MTA website is up-to-date traffic generated by real malware samples;
- every dataset in Booters contains over 100 attack logs, thus these are representative examples of DDoS attacks;
- CTU-13 contains, among other things, non peer-to-peer traffic, which is not a common feature among other datasets;
- the ISOT dataset is already a blend of four other datasets, and contains, among other things, normal traffic generated by a variety of everyday-use applications (such as games), which is an added value.

Therefore we claim that the resulting dataset, created from such a diversity of sources, will be truly heterogeneous, and thus will pose a challenge when designing algorithms for attack detection.

4.2 Time alignment

Datasets used in VHS-22 contain at least one pcap file; each of those files was converted to a file with flows. For time-alignment purposes, a reference date has been chosen. Subsequently, the file contents of the first file with flows were altered in terms of date to fit every packets' time signature in one day. Each file with flows has its own day, starting from the reference date (1 Jan 2022). Exact timespans along with the source datasets are shown in Table 1.

4.3 Extracted network traffic parameters

Table 2 displays the list of parameters collected during traffic data analysis. They contain basic *flow-level* information such as source and destination IP addresses, source and destination ports, protocols, length of flow expressed in packets, and timestamps of the first and last packet in a flow. What is not so common is that information about interarrival time (IAT) between packets is also extracted. We are aware that some of these parameters are correlated (e.g., *in_packets* and *duration*), nevertheless we decided to keep them in our dataset, to give freedom of choice to future users of the dataset. These data are accompanied by statistical parameters, such as means, standard deviations, variances, first and third quantiles of packet size, and IAT. In addition, each TCP flow contains information about URG, ACK, RST and SYN flags, and their sum (as a separate parameter).

All the flows produced by network probe were post-processed. This operation allowed for extraction of additional *network-level* parameters to enable anomaly detection on the network plane (i.e., taking into account other flows). They include features such as the total number of connections to/from a given host, or ratios of connection to/from a given host against the total number of active flows.

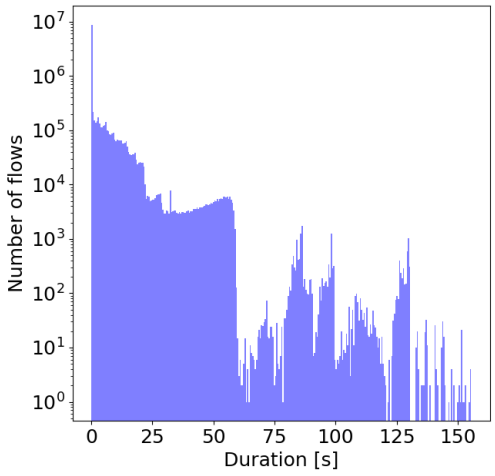

Figure 2: Histogram of flow duration in VHS-22 (values for flows over 160 sec, constituting 0.01% of TCP flows, are not displayed)

Each flow is labeled with the class label (0 – *normal*, 1 – *attack*), the attack label, and the source dataset label. The labels for 45 flow parameters formed the VHS-22 dataset, and were stored in the .csv format.

4.4 Dataset statistics

We have calculated the basic statistics of the created VHS-22 dataset. Table 3 displays the count of flows, both for normal and attack-related traffic. The VHS-22 dataset contains more than 27 million flows, out of which roughly 20 million represent regular traffic and

Table 2: Summary of extracted flow parameters available in VHS-22 (flow-level features in upper part and network-level ones in lower part).

Parameter	Description
ip_src_str	source IP address
ip_dst_str	destination IP address
ip_protocol	fourth layer protocol
sport	source port
dport	destination port
in_packets	sum of packets in flow
b_packet_total	sum of bytes in flow
first_timestamp	first timestamp in flow
last_timestamp	last timestamp in flow
duration	duration of flow
flags_sum	sum of TCP flags (URG-32, ACK-16, PSH-8, RST-4, SYN-2, FIN-1)
urg_nr_count	URG flag count in flow
ack_nr_count	ACK flag count in flow
rst_nr_count	RST flag count in flow
fin_nr_count	SYN flag count in flow
psh_nr_count	PSH flag count in flow
syn_nr_count	SYN flag count in flow
b_packet_max	size of the largest packet
b_packet_min	size of the smallest packet
b_packet_mean	mean packet size
b_packet_median	median packet size
b_packet_first_q	1^{st} quantile of packet size
b_packet_third_q	3^{rd} quantile of packet size
b_packet_std	std. deviation of packet size
b_packet_total	total size of flow in bytes
iat_min	lowest IAT
iat_max	highest IAT
iat_first_q	1^{st} quantile of IAT
iat_third_q	3^{rd} quantile of IAT
iat_std	standard deviation of IAT
iat_mean	mean of IAT
iat_median	median of IAT
iat_var	variance of IAT
connections_from_this_host	No. of connections from given host
connections_to_this_host	No. of connections to given host
connections_rst_to_this_host	No. of connections to host ended with RST flag
connections_rst_from_this_host	No. of connections from host ended with RST flag
connections_to_this_port	No. of connections with the same destination port number
connections_from_this_port	No. of connections with the same source port number
connections_ratio_from_this_host	% of connections to the host with the same destination address
connections_ratio_to_this_host	% of connections from the host with the same source address
connections_ratio_rst_to_this_host	% of connections to host ended with RST flag
connections_ratio_rst_from_this_host	% of connections from host ended with RST flag
connections_ratio_to_this_port	% of connections to host with the same destination port
connections_ratio_from_this_port	% of connections from host with the same source port

about 7 million represent network attacks. In both of these groups we identified flows which contain only one packet, which we refer to as zero-duration flows. They constitute 45% of normal traffic and 83% of attacks (mostly DoS-related).

The protocol statistics are shown in Table 4. It turned out that about 62% of flows belong to the UDP protocol, while the remainder are TCP. Zero-duration flows constitute 66% and 38% of UDP and TCP protocol flows, respectively.

Table 3: Flow count in VHS-22.

Normal traffic		**Attacks**	
Dur. > 0	Dur. = 0	Dur. > 0	Dur. = 0
11 189 118	9 151 283	1 247 017	6 148 257
20 340 401		7 395 274	
Total: 27 735 675			

Table 4: VHS-22 protocol statistics.

TCP flows		**UDP flows**	
Dur. > 0	Dur. = 0	Dur. > 0	Dur. = 0
6 544 911	4 017 530	5 891 224	11 282 010
10 562 441		17 173 234	
Total: 27 735 675			

Table 5: VHS-22 attack statistics.

Attack	Flows number
DoS	6 778 273
Botnet	520600
MTA Traffic	85331
Brute force	7022
Web	4048

Figures 2 and 3 display histograms of flow-duration values and flow length. Both these parameters reach the highest values on the left side of the histograms, which correspond to the shortest, one-packet flows. The duration histogram (Figure 2) omits very long flows, i.e., lasting longer than 160 sec. One notices that the vast majority of flows last between 0 and 60 sec. The dataset also contains flows longer than 160 sec, originating from the ISOT dataset. Their lengths concentrate around 33 min, 1h 43' and 2h 15'.

The flow length decreases approximately exponentially (Figure 3). The majority of flows are shorter than 200k packets, however, the longest one is almost 11.7M packets long.

The attack distribution, shown in Table 5, reveals that the highest number of network flows labeled as 'attacks' is caused by various DoS-type and Botnet-related attacks (93.9% and 5.8%, respectively). The remaining attack-related traffic was originated by various types of malware (0.15%), brute force attacks (0.8%) and web attacks (0.05%).

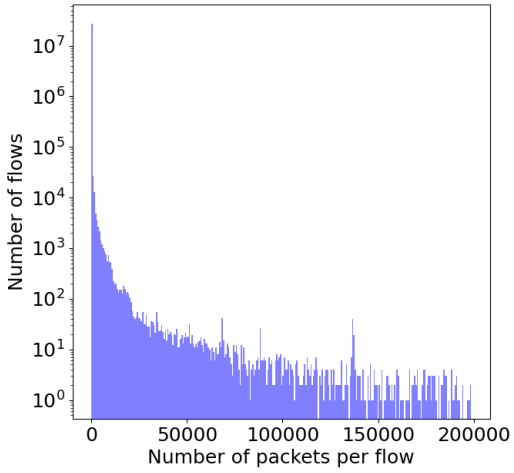

Figure 3: Histogram of flow length (in packets) in VHS-22 (flows longer than 200k packets, constituting 0.01% of all flows, are not displayed)

		Predicted	
		Legitimate Traffic	Attack Traffic
Actual	Legitimate Traffic	True Negative (TN)	False Positive (FP)
	Attack Traffic	False Negative (FN)	True Positive (TP)

Figure 4: Confusion matrix with possible error types for attack detection

5 INITIAL EXPERIMENTS

To set a baseline, we ran a series of initial experiments on detecting attacks in the VHS-22 dataset. The experiment methodology and results are briefly presented in this section.

5.1 Evaluation metrics

To evaluate the detection performance, we defined typical metrics, based on binary classification errors explained in Figure 4.

- Accuracy – the fraction of correct predictions, expressed as:

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN} \tag{1}$$

- Precision Score – the ability of the classifier not to label as an attack a sample that is normal:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

- Recall Score – the ability of the classifier to find all the attack flows:

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

- F1 Score – the harmonic mean of the precision and recall:

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4}$$

Table 6: Detection results for VHS-22 compared with results for other datasets (dashes mean no data were available).

Dataset	Algorithm	Accuracy	Precision	Recall	F1-Score	Source
VHS-22	SGD	0.88	0.98	0.52	0.68	This study
(flow-level features)	RF	0.96	0.94	0.89	0.91	
VHS-22	SGD	0.92	0.96	0.72	0.82	This study
(flow- and network-level features)	RF	0.97	0.97	0.93	0.95	
CICIDS 2017	SGD	0.98	0.97	0.94	0.96	This study,
	RF	0.99	0.99	0.99	0.99	same conditions
CICIDS 2017	*k*-NN	–	0.96	0.96	0.96	[16]
	RF	–	0.98	0.97	0.97	
	ID3	–	0.98	0.98	0.98	
	AdaBoost	–	0.77	0.77	0.77	
	QDA	–	0.97	0.92	0.92	
UNSW-NB15	AdaBoost	0.86	0.87	0.93	0.90	[2]
	MFFANN	0.99	–	–	–	
	RepTree	0.89	–	–	–	
	Naïve Bayes	0.88	–	–	–	
CTU-13	NNM	0.98	0.86	0.84	0.85	[3]
	RNNM	0.83	0.95	0.69	0.80	
KDD-MTA 19	MLP	0.99	0.99	0.99	0.99	[9]
RoEduNet–SIMARGL2021	RF	0.99	0.99	0.99	0.99	[10]
	AdaBoost	0.54	0.43	0.54	0.47	
	GBT	1.00	1.00	1.00	1.00	
	DNN	1.00	0.99	1.00	0.99	
ISOT	*k*-NN	0.99	0.99	0.99	0.99	[1]
	RF	1.00	1.00	1.00	1.00	

5.2 Testing methodology

The flows from VHS-22 were divided into training and testing subsets in the proportion 70 : 30 from each day. Stochastic gradient descent (SGD) and random forests (RF) from the scikit-learn library [17] were employed as the binary classifiers. For RF, the number of trees in the forest was set to 100 and the Gini impurity was used as the measure of split quality. For SDG, a linear SVM was employed as the loss function. The detection was realized using either flow-level features or in a combined, flow- and network-level feature space. The classification results are described in the next section.

5.3 Initial results

Table 6 presents the results of initial attack detection experiments run on the VHS-22 dataset, compared to other studies. One easily notices that the results achieved are lower than those reported by researchers for other datasets. When using flow-level features, in the best case, for the RF classifier we managed to achieve the F1-Score at the level of 91%. After adding network-level features (which, by the way, are not always available for a detection algorithm), this score increased to 95%. Using the same testing conditions (the same classifiers, their parameters, and the feature space) we achieved 99% for CICIDS 2017. The other studies on the other datasets (including those constituting VHS-22) the best reported values oscillate between 85% and 100%, often being close to the latter score.

6 CONCLUSIONS

In the context of ML-aided network flow classification, using homogeneous datasets easily leads to overfitting of the detection algorithms. In our work we present a much more heterogeneous flow-level network traffic dataset, named VHS-22, which is open to the research community and available, with the license information, at the address https://www.kaggle.com/datasets/h2020simargl/vhs-22-network-traffic-dataset. Initial detection results showed that detection of attacks is still a challenge. Using such a dataset may foster the development of more efficient and more universal detection methods for network attacks.

ACKNOWLEDGMENTS

This study has been funded by the SIMARGL Project – Secure Intelligent Methods for Advanced RecoGnition of malware and stegomalware, with the support of the European Commission and the Horizon 2020 Program, under Grant Agreement No. 833042.

REFERENCES

[1] Amirah Alshammari and Abdulaziz Aldribi. 2021. Apply machine learning techniques to detect malicious network traffic in cloud computing. *Journal of Big Data* 8 (06 2021). https://doi.org/10.1186/s40537-021-00475-1

[2] AvinashR.Sonule, Mukesh Kalla, Amit Jain, and Deepak Singh Chouhan. 2020. UNSW-NB15 Dataset and Machine Learning Based Intrusion Detection Systems. *International Journal of Engineering and Advanced Technology* 3, 9 (2020), 2638–2648.

[3] Ankit Bansal and Sudipta Mahapatra. 2017. A Comparative Analysis of Machine Learning Techniques for Botnet Detection. In *Proceedings of the 10th International Conference on Security of Information and Networks* (Jaipur, India) *(SIN '17)*. Association for Computing Machinery, New York, NY, USA, 91–98. https://doi.org/10.1145/3136825.3136874

[4] Brad. 2022. Malware Traffic Analysis. https://www.malware-traffic-analysis.net

[5] Ayesha S. Dina and D. Manivannan. 2021. Intrusion detection based on Machine Learning techniques in computer networks. *Internet of Things* 16 (2021), 100462. https://doi.org/10.1016/j.iot.2021.100462

[6] The Tcpdump Group. 2022. Libpcap Library. https://www.tcpdump.org/.

[7] Riaz Khan, Xiaosong Zhang, Rajesh Kumar, Abubakar Sharif, Noorbakhsh Amiri Golilarz, and Mamoun Alazab. 2019. An Adaptive Multi-Layer Botnet Detection Technique Using Machine Learning Classifiers. *Applied Sciences* 9 (06 2019), 2375. https://doi.org/10.3390/app9112375

[8] McAfee Labs. 2016. McAfee Labs 2016 Threats Predictions. https://www.intel.com/content/dam/www/public/us/en/documents/reports/mcafee-2016-threats-and-predictions-report.pdf

[9] Ivan Letteri, Giuseppe Penna, Luca Vita, and Maria Grifa. 2020. MTA-KDD'19: A Dataset for Malware Traffic Detection. In *Proc. Fourth Italian Conference on Cyber Security (ITASEC 2020)*. CEUR, Ancona, Italy, 153–165.

[10] Maria-Elena Mihailescu, Darius Mihai, Mihai Carabas, Mikołaj Komisarek, Marek Pawlicki, Witold Hołubowicz, and Rafał Kozik. 2021. The Proposition and Evaluation of the RoEduNet-SIMARGL2021 Network Intrusion Detection Dataset. *Sensors* 21, 13 (2021), 4319. https://doi.org/10.3390/s21134319

[11] Nour Moustafa and Jill Slay. 2015. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *Proc. Military Communications and Information Systems Conference (MilCIS 2015)*. IEEE, Canberra, Australia, 1–6. https://doi.org/10.1109/MilCIS.2015.7348942

[12] Markus Ring, Sarah Wunderlich, Deniz Scheuring, Dieter Landes, and Andreas Hotho. 2019. A Survey of Network-based Intrusion Detection Data Sets. *Computers & Security* 86 (2019), 147–167. https://doi.org/10.1016/j.cose.2019.06.005

[13] Sherif Saad, Issa Traore, Ali Ghorbani, Bassam Sayed, David Zhao, Wei Lu, John Felix, and Payman Hakimian. 2011. Detecting P2P botnets through network behavior analysis and machine learning. In *Ninth Annual International Conference on Privacy, Security and Trust*. IEEE, Montreal, Canada, 174–180.

[14] José Jair Santanna, Romain Durban, Anna Sperotto, and Aiko Pras. 2015. Inside booters: An analysis on operational databases. In *International Symposium on Integrated Network Management (IM 2015)*. IFIP/IEEE, Ottawa, Canada, 432–440. https://doi.org/10.1109/INM.2015.7140320

[15] Iman Sharafaldin., Arash Habibi Lashkari., and Ali A. Ghorbani. 2018. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy - ICISSP,*. INSTICC, SciTePress, Portugal, 108–116. https://doi.org/10.5220/0006639801080116

[16] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. 2018. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proc. 4th International Conference on Information Systems Security and Privacy (ICISSP 2018)*. INSTICC, SciTePress, Funchal, Portugalia, 108–116. https://doi.org/10.5220/0006639801080116

[17] Scikit-Learn Open source comunity. 2013. Scikit-Learn. Http://scikit-learn.org/stable/.