A review of neural networks for rare intrusions detection in wireless networks

Vu Viet Thang, Dmitry Valerievich Pantiukhin, Bui Thi Thanh Quyen, Vu Viet Vu

***Abstract* — Neural networks have become the most popular approach for detecting tasks. Currently, neural networks have been strongly applied in the fields of image processing, text and signal processing and have achieved certain effectiveness. However, they have not been widely applied in information security and intrusion detection. Especially, there are no much applications of neural network about rare attacks. In our review, the rare attacks are attacks with a low number of instances or unfamiliar types of security attacks with a low occurrence rate. This is due to a lack of labeled data required for neural networks training and a significant imbalance in the number of different data classes. In this article, we have researched, compared and evaluated current methods for solving problems such as data augmentation, data generation via generative networks and classing importance control. Additionally, we will also provide a brief overview of existing datasets for intrusion detection in wireless networks.**

***Tóm tắt* — Mạng nơ-ron đã trở thành phương pháp phổ biến nhất trong bài toán phát hiện đối tượng. Hiện nay, mạng nơ-ron đã được ứng dụng mạnh mẽ trong các lĩnh vực xử lý hình ảnh, xử lý văn bản, tín hiệu và đạt hiệu quả nhất định, tuy nhiên mạng nơ-ron chưa được áp dụng rộng rãi trong bảo mật thông tin và phát hiện xâm nhập. Đặc biệt, chưa có nhiều các nghiên cứu về việc phát hiện các cuộc tấn công hiếm gặp. Các cuộc tấn công này được hiểu là các dạng tấn công có số lượng ít hoặc ít phổ biến trên thực tế so với các dạng tấn công khác. Điều này dẫn tới việc thiếu dữ liệu được gán nhãn cho việc huấn luyện mạng nơ-ron và sự mất cân bằng lớn của các lớp dữ liệu khác nhau. Trong bài báo này, nhóm tác giả đã đi tìm hiểu, so sánh và đánh giá về các phương pháp hiện tại trong việc giải quyết các vấn đề như làm giàu dữ liệu, sinh dữ liệu thông qua mạng sinh (generative networks) và các vấn đề liên quan khác. Ngoài ra, nhóm tác giả còn trình bày tổng quan về các bộ dữ liệu phổ biến hiện nay để phát hiện xâm nhập trong mạng không dây.**

Keywords— Rare attack; intrusion detection; neural network; generative network; wireless network.

Từ khóa— Tấn công hiếm gặp; phát hiện xâm nhập; mạng nơ-ron; mạng sinh; mạng không dây.

I. INTRODUCTION

Nowadays, machine learning is widely applied and becoming more and more popular. Its most significant achievements have been in image processing where convolutional neural networks play a crucial role. However, there are many researchers who have tried applying these methods in other areas such as information security, specifically in intrusion detection and classification. Then, they have tried using neural networks to solve the problem of intrusion classification.

However, we encounter a problem: the number of examples for each intrusion class varies significantly, and the classifier must be trained with this.

Researchers can develop various intrusion classifiers, based on different approaches. Some existing approaches include:

- Signatures of intrusions [1].
- Protected system state analysis [2].
- Scenario graphs and attack graphs [3].
- Expert systems [4].
- Multivariate adaptive regression splines [5].
- Support vector machines [6].
- Immune networks and genetic algorithms [7].
- Neural networks, including recurrent, convolutional and others [8].

This manuscript is received on March 08, 2023. It is commented on April 19, 2023 and is accepted on June 07, 2023 by the first reviewer. It is commented on May 08, 2023 and is accepted on June 05, 2023 by the second reviewer.

Neural networks are one of the most popular techniques to solve detection tasks in image, voice, and text. They are also used for intrusion detection in both wired and wireless networks. However, there is a significant problem in applying this technique for rare intrusions detection. Neural networks require a large amount of data for training, and it can be difficult to collect enough examples for effective training in the case of rare intrusions. As a result, we have not yet seen the same level of success in intrusion detection as we have in image detection. To address this problem, various approaches have been taken from other areas and have shown usefulness in the field of intrusion detection.

Intrusion detection systems can be classified as host-based or network-based, depending on the source of information. The latter can be divided into wired or wireless systems, depending on the structure of the protected network. For the purpose of this study, we will focus on network-based systems, both wired and wireless. Another classification is based on the detection approach: misuse detection, whose goal is to build a model of exact intrusions and anomaly detection, whose goal is to build a model of normal behavior and detect any deviations from it. Both approaches are valuable, but they require a significant amount of data to train the neural network for modeling and are subject to the issue of class imbalance.

II. DATASETS

The most popular intrusion dataset for a long time was the KDD99'Cup [9], which has been available since 1998-1999. This open and free-to-use dataset was created by the Information and Computer Science University of California. It consists of approximately 5 million records about network transactions, each record including 41 parameters of network traffic such as destination IP\port and source IP\port. These parameters are divided into three types: categorical, logical (flags) and numeric. The dataset contains information about 22 types of intrusion, which are further divided into four main classes: Denial of Service (DoS, 3883370 records), Remote to User (R2L, 1126 records), User to Root (U2R, 52 records), Probe (41102 records), and one class of Normal (972780 records) packets. The number of records of each class is quite different. Despite being criticized [11], this dataset remains the most popular-one. Its main disadvantages include:

- A large number of duplicated, redundant records (about 80%!),

- Lack of records of some intrusion classes (U2R, R2L),

- Moral obsolescence – the database was collected in 1998-1999 and did not contain information about modern attacks.

In 2009, ten years later, KDD99Cup was modified to become NSL-KDD [9], redundant records were dropped, some work for refining was done, and the dataset was reduced four times.

It can also be mentioned that dataset the ADFA from the Australian Defense Force Agency operates on a different principle. Instead of containing records of network traffic, it consists of information about running processes on a host machine. This makes it suitable for use in host-based intrusion detection systems, which is not within the scope of our current work.

In 2023, the CICIOT dataset [17] was created at the Canadian Institute for Cybersecurity (CIC). E. C. P. Neto and el. Performed, documented, and collected data from 33 attacks divided into 7 (Benign, Brute Force, DdoS, DoS, Mirai, Recon, Spoofing, web) classes against IoT devices. However, there is an unbalancing data between class "benign" and "attack categories". The authors conducted experiments applying machine learning models on this dataset. The result showed that some classes are very well classified, mainly those with a large number of occurrences in the dataset. For example, the misclassification rates for DDoS, DoS, and Mirai are very small, followed by Recon and spoofing. This once again proves that imbalancing data is very harmful to classification quality.

TABLE 1. DATASETS FOR INTRUSION DETECTION

Name	Year	Wired or wireless	# of classes	# of normal class examples (approx)	Imbalanced ratio
NSL KDD [9]	2009	Wired	5	0.06 M	250
Kyoto 2006+ [10]	2011	Wired	3	50 M	100
TUIDS [11]	2012	Wired	3	0.07 M	10
OTIDS [12]	2017	Wired	4	2.3 M	4
LITNET-2020 [13]	2020	Wired	13	~40 M	2500
AWID2 \AWID3 [14]	2016, 2021	Wireless	13 (21)	~35 M	1500
WSN-DS [15]	2016	Wireless	5	0.2 M	100
IoT [16]	2019	Wireless	4	1.7 M	100
IoT [17]	2023	wired	8	0.24 M	49

The area of intrusion detection has long suffered from a lack of publicly available datasets. Indeed, the DARPA KDD99Cup (modified to NSL KDD [9] in 2009) dataset has been the only widely known one since 1999. However, there are now a growing number of datasets available for both wired and wireless networks. But all of these datasets are affected by the class imbalance problem, as shown in Table 1, the imbalance ratio, which is the ratio of the number of normal class examples to the rarest class (intrusion) (approx.) examples, is provided in the table. It should be noted that the term class refers to a group of intrusions that are typically detected as one class, even though the datasets themselves may have a more detailed hierarchy of intrusions. As demonstrated in [18] most of these datasets are not balanced.

III. DATA BALANCING APPROACHES

Then, try applying the neural network or the other data-driven technique as intrusions classifier data imbalance plays an important role and restricts accuracy for rare classes. Therefore, special approaches are required to address imbalanced datasets. We can describe such approaches in 3 groups: data reduction, data augmentation and imbalance accounting.

First, data reduction approaches suggest removing some data from non-rare classes to correctly balance them with rare classes. For example, we can use the K-means method to cluster data samples from non-rare classes and replace a large number of data samples with a smaller number of their centroids.

Second, data augmentation suggests various methods for creating and generating new synthetic data for rare classes.

Lastly, the imbalance accounting approach suggests using information about the number of class examples during training. This can be achieved by modifying the loss function with additional weights for different classes (giving a higher weight to the rarer class) or by controlling the number of class examples in each training batch.

A. Online batch balancing control

While a neural network is trained batchwise, we have the ability to control the number of examples from different classes in each batch on the fly. Instead of randomly selecting examples from the entire dataset, we can specify a predetermined number of examples from each class to be included in the batch (Figure 1). For example, if the batch size is 300 and we have 10 classes, we can create a batch that contains 30 random examples from each class. It means that data from rare classes are repeated during training frequently.

In our experiments [19] we utilized a convolutional neural network (Figure 2) as an intrusion classifier for the UNSW-NB15 dataset.

It consists of 5 convolutional layers with ‘sigmoid’ activations and 3 fully connected layers, first and second with ‘sigmoid’ and last with ‘softmax’ activation. Number of inputs – 190, number of outputs – 10 (number of classes, 9 intrusions and Normal).

To use the chosen dataset, we need to preprocess data to be suitable for neural networks. We made it in three stages: choosing relevant features, coding non-numeric data and scaling numeric data.

Firstly, we need to drop some irrelevant features. The UNSW-NB15 has 47 feature fields, some of which relate only to the computer network used for dataset collection. Namely, we drop the following 7 fields from the dataset:

- ‘srcip’ (source IP address)
- ‘dstip’ (destination IP address).
- ‘sport’ (Source port number).
- ‘dsport’ (destination port number).
- ‘stime’ (record start time).
- ‘ltime’ (record last time), and ‘res_bdy_len’ because this field does not change in dataset.

- Among other fields, the dataset contains 3 categorical (string) fields:

- ‘proto’ (transaction protocol), has 129 different values (‘udp’, ‘tcp’, arp’ etc.).

- ‘servis’, has 8 values (‘http’, ‘ftp’, ‘smtp’, ‘ssh’, ‘dns’, ‘ftp-data’, ‘ir and '-‘ if not used).

- ‘state’, depend on transaction protocol and has 16 values (ACC, CLO, CON, ECO, ECR, FIN, INT, MAS, PAR, REQ, RST, TST, TXD, URH, URN and ‘-‘ if not applicable).

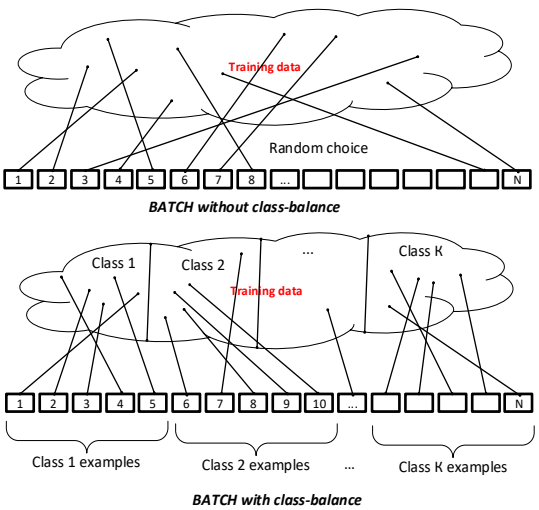

Figure 1. Variants of batch forming. Top – random selection, bottom – class-balanced selection

B. *Neighbors’ interpolation*

We can create new synthetic examples of rare classes to increase the number of examples. The Naive approach is just to repeat data, but we can go further and create rare classes of new examples randomly by interpolation of attributes between K-neighbors of real examples. Such an approach to balancing was realized in [20] and is known as the Mean Synthetic Minority Oversampling Technique (M-SMOTE). In [21] they combine SMOTE with the OSS (One-side selection) that rejects noise and redundant data from non-rare classes and reaches a balanced dataset. After applying a balanced dataset with a deep hierarchical neural network classifier (that combines convolution and recurrent networks) they reach the F1 metric of ~35% for the rarest class (U2R), while maintaining ~92% for normal non-rare class on the NSL-KDD dataset and ~11% (worms)\ 85% (normal) on the UNSW-NB15 dataset.

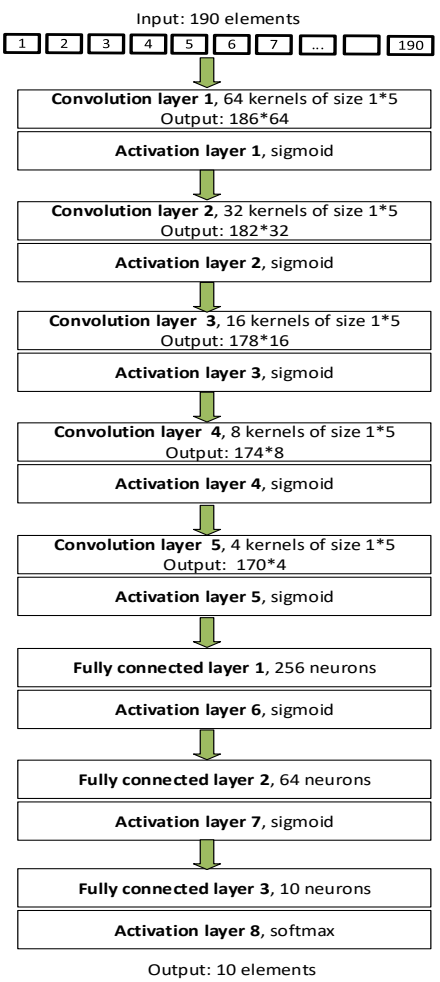

Figure 2. Structure of used neural network

Modification of SMOTE ideas was implemented in the Difficult Set Sampling Technique [22] where datasets are easy and difficult classification sample sets: If for some samples among its K-neighbors, there exist neighbors of another class, they are marked as difficult. The easy set is not changed. For difficult sets, for example, majority class samples (non-rare) are compressed to their centroids using KMeans clustering, and minority class samples (rare) are zoomed: creating new samples in which discrete attributes remain unchanged, but continuous attributes change (scaled) in the interval [1-1\K, 1+1\K]. Applying such a balancing approach for training miniVGGnet on the CSE-CIC-IDS2018 dataset (imbalance ratio up to 800), they reach the F1 metric of 0.97, while the same network with SMOTE balancing gives the F1 of 0.93.

C. Generative networks for synthetic examples creating

Attribute distributions between intrusions may be complex. Therefore, it is not enough to use simple methods like interpolation to keep correct distributions.

Given the samples from different classes, we can estimate the attribute distribution and resample randomly by using the same distribution. Such an approach with the kernel density estimation method was implemented to create a dataset of network traffic.

To deal with complex distributions, it is possible to use generative-adversarial networks (GAN). GAN consists of two networks: a generator and a discriminator. The discriminator tries to distinguish between real and synthetic samples, while the generator attempts to generate samples, which will confuse the discriminator. By training GAN in a competitive way, it is possible to create a generator that can generate samples which look like real ones. Such an approach was implemented in [23], where they use the AC-GAN modification to create samples of SSH-like data to augment the NIMS dataset (SSH and nonSSH traffic with a ratio of 1:20) and reach an accuracy of more than 99%.

Many GAN modifications can be applied to increase generation performance. In [24], the authors introduce a Bidirectional GAN for intrusion detection. The basic GAN structure suggests that the generator maps latent vector space to the real space (space of real data). In Bi-GAN, the inverse process is also applied – the encoder part of the generator learns the inverse mapping of real samples to the latent space, giving control in the latent space. The discriminator uses both representations in the latent and the real spaces to make a decision. This approach only deals with binary classification (normal\intrusion) and is trained on normal samples. Anything that is not recognized as normal is treated as an intrusion.

In other research [25], Author Andrei-Grigore Mari and his colleagues used the GAN approach to improve the machine learning-based IDS performance. This paper delves into the impact of adversarial data generated through a GAN on an Intrusion Detection System (IDS) that incorporates machine-learning algorithms. Initially, we implemented the IDS and showcased its performance with each algorithm. Following that, a GAN was implemented wherein the discriminator employed one of the previously tested algorithms, and the generator utilized the same algorithm, aiming to maximize the discriminator's loss. In comparison with the original performance (when tested with the NSL-KDD dataset), the performance in the presence of adversarial traffic is higher in most machine-learning techniques.

IV. Analysis of existing systems for detecting computer attacks in local wireless networks: advantages and disadvantages

A. Intrusion Detection Methods

There are two main methods for detecting threats: signature-based detection and anomaly-based detection. These methods come in handy when the user is deciding whether to use the signature or anomaly detection mechanism. However, IDS service has already recognized the benefits of each method and are now attempting to incorporate both into their products. By examining the strengths and weaknesses of these

methods, we can better understand how they can complement each other [26].

Signature-based IDS tools use specified rules or patterns to search for known malicious traffic. When a match is found, a warning about a possible threat will be sent to the administrator. These alerts can report issues such as malware, fraudulent network scans, and server attacks.

For anomaly-based IDS Tools, the activity that leaves behind traffic is more important than the payload being delivered. These tools rely on baselines, not signatures, and look for unusual activity that is considered a deviation from the statistical averages of previously seen activity. For example, if a user typically logs in from California accesses to working files, but someone tries to log in from Beijing and view HR files, this would be considered a threat. Both signature and anomaly detection methods are usually deployed in the same manner. However, it is possible to generate anomalous identifiers based on externally collected Netflow data or similar traffic information.

TABLE 2. GROUPS AND TYPES OF ATTACKS ON WIRELESS LANS

Group	Types
Passive attacks	FMS, Korek, PTW, dictionary attack (Dictionary)
Flooding	Deauthentication, Disassociation, Deauthentication broadcast, Disassociation broadcast, Block Acknowledge, Authentication Request, Fake Power Saving, CTS, RTS, Beacon, Probe Request, Probe Response
Injection	ARP-Injection, ChopChop, Fragmentation
Middleman attacks (Impersonation)	Honeypot, Evil Twin, Caffe Latte, Hirte

Fewer false messages are generated by signature-based detection, but this method only flags known signatures, leaving security holes for new and unidentified threats. On the other hand, many false notifications can occur when anomalies are detected, but if the tool is configured correctly, it can also catch previously unknown threats.

Network-Based IDS (NIDS) inspect all traffic on the network segment to detect malicious activity. This is achieved by delivering a copy of the traffic to the NIDS device which then monitors and alerts for traffic patterns or signatures. When a malicious event is detected, the sensitive information is logged into the program. Data needs to be tracked to know what's going on in the system. By combining this information with events collected from other systems and devices, a complete picture of the network's security can be constructed. It is worth noting that none of the tools mentioned in this article can correlate journals on their own. This is typically a function of a SIEM.

As a result, the main idea of intrusion detection methods, as applied to an intrusion detection system (IDS), is based on the idea that the activity of users and programs on the network can be monitored and a mathematical model can be built.

Methods of data mining for detecting attacks in IDS can be divided into two groups:

- Misuse detection methods, which allow for the construction of a model of an attack, and in the process of detection they use data mining methods for classification.

- Anomaly detection methods, which build a model of normal activity, and in the process of detection use the data mining methods to identify exceptions.

There are the following methods used to detect anomalies:

- Statistical methods
- Methods based on classification
- Clustering methods
- Methods based on K-nearest neighbors

We will follow existing systems on these grounds as shown above approaches, methods, and algorithms for intrusion detection. Consider

the following existing IDS systems: Snort, Suricata and Bro (Zeek).

B. Snort

Snort is a free and open-source software licensed under the GPL. It was initially created in 1998 by Martin Roche, a well-known figure in the world of information security and the author of many books. The main reason for the creation of this IDS was the absence at that time of a sufficiently effective, especially free, attack notification tool [27]. Snort can detect the following:

- Bad traffic.

- Use of exploits (detection of Shell code).

- System scan (ports, OS, users, etc.).

- Attacks on services such as Telnet, FTP, DNS, etc.

- DoS/DDoS attacks.

- Attacks related to Web servers (cgi, php, frontpage, iss, etc.).

- Attacks on SQL databases, Oracle, etc.

- Attacks via SNMP, NetBIOS, ICMP protocols.

- Attacks on SMTP, imap, pop2, pop3.

- Various Back doors.

- Web filters.

Barnyard2 is an open-source interpreter for binaries generated by Snort. The standard way of recording events provided by Snort to the console or a file can be resource - intensive. Ideally, Snort events should be stored in a MySQL database so that you can find and view the events you want. Barnyard2 will be used to make Snort events appear in the MySQL database. Snort is configured to output binary events to a folder, and then Barnyard2 reads these events and inserts them into the MySQL database.

PulledPork is a script that, automates the process of downloading, combining, configuring, and updating rules for Snort from various sources. It can also be configured to download the free Snort community ruleset without creating a free Snort.org account.

BASE (Basic Analysis and Security Engine) is a basic analysis and security engine. This program is needed to visualize detected attacks. It simplifies the work with logs, displays and stores event logs in a convenient format [28].

C. Suricata

Like Snort, Suricata consists of several modules (capturing, collecting, decoding, detecting and outputting). By default, before decoding, the captured traffic is processed in one stream. While this is optimal for detection, it can also put a heavier load on the system.

Unlike Snort, in Suricata you can redefine the traffic behavior with settings. With just one setting in the configuration, the stream can be split immediately after capturing. Additionally, another setting allows for the distribution of streams among the processors. This provides ample opportunities for optimizing traffic processing on specific equipment in a specific network. Suricata also supports the extraction and verification of files transmitted over HTTP, parsing of compressed content, and the ability to identify by URI, cookie, headers, user-agent, request, and response body. This feature is often used in some networks to log HTTP traffic without detection. The content in the stream can be distinguished using masks and regular expressions, and files can be identified by name, type, or MD5 checksum [29].

TABLE 3. COMPARISON OF WLAN MISUSE DETECTION ALGORITHMS

Algorithm	**Advantages and Benefits**	**Disadvantages**
Neural Networks	- Ability to generalize unstructured incomplete data - No need for expert knowledge - Ability to detect new intrusions	- Slow learning process, therefore not suitable for real-time detection - Overtraining may occur
Bayesian Network	- Encodes the probabilistic relationship between variables of interest.	- Harder to handle continuous dependencies - may not give a qualitative

	- The ability to use prior knowledge	classification if prior knowledge is erroneous
Support Vector Machine	- Good learning for a small number of examples - High learning rate and speed of decision - making - Insensitivity to the dimension of the input data	- Training takes a long time - Mostly a binary classifier is used, which cannot provide additional information about the type of intrusion detected.
Genetic Algorithm	- The ability to develop the best classification rules and select the optimal parameters	- Cannot provide constant response time. - Overtraining may occur.
Fuzzy Logic	- Fuzzy rules are more understandable to humans - Especially effective against port scans and probes	- High computational complexity - Simplified identification of the appropriate subset of rules and dynamic update of rules at runtime is challenging.
Decision Tree	- Works well with large amounts of data - High detection accuracy	- High computational complexity.

IPv6 decoding is natively supported, including tunnels IPv4-in-IPv6, IPv6-in-IPv6, Teredo, and some others. The modular layout of the engine makes it possible to quickly connect a new element for capturing, decoding, analyzing or processing packets. To intercept traffic, several interfaces are used - NF Queue, IPF Ring, Lib Pcap, IPFW, AF_PACKET, PF_RING. Unix Socket mode allows you to automatically analyze PCAP files previously captured by another program (sniffer, for example).

TABLE 4. COMPARISON OF ANOMALY DETECTION METHODS FOR WIRELESS LOCAL NETWORK

Method	**Advantages and Benefits**	**Disadvantages**
Statistical methods	- Ability to detect new, yet unknown attacks (zero-day) - Does not require any prior knowledge - No need to configure for a specific system- attackers can present abnormal behavior as normal - Difficult to control false positive and negative positives	- Attackers can present abnormal behavior as normal - Difficult to control false positive and negative positives
Clustering methods	- Easy-to-implement algorithms - Low computational complexity - No need for labeled data - The ability to detect new and unknown anomalies	- Provide good results only for spherical clusters of the same volume - Difficult to apply to datasets of complex shapes
Methods based on classification (neural networks, Markov model)	- Easy-to-implement algorithms - The ability to adapt to a specific task - hHgh classification accuracy - Quick classification after training	- Takes time to learn - Requires tagged data - High computational complexity of training - Difficult to recognize unknown anomalies
Methods based on K-nearest neighbors (LOF, LOCI, etc.)	- No prior knowledge required - Works well with complex shapes and different data densities - Can recognize new and unknown anomalies with high accuracy	- High difficulty - Many false positives

In Snort, the IPS mode is not immediately available. However, in Suricata, the malicious traffic blocking mode is implemented out of the box and can be performed using the standard OS packet filter. Linux, for example, uses two IPS modes: through the NFQUEUE queue, which can be processed at the user level, and through the zero - copy AF_PACKET mode (introduced in version 1.4). The AF_PACKET mode is very fast, but it requires two network interfaces and the system must work as a gateway. If a packet is blocked, it simply isn't forwarded to the second interface. In the case of NFQUEUE, the algorithm is simple. After a packet enters iptables, it is sent to the NFQUEUE queue, where it is run according to the rules. There are three result actions: NF_ACCEPT, NF_DROP, and NF_REPEAT. The NF_REPEAT allows you to mark packets and then run them according to the given tables/rules [29].

The main feature of Suricata is that in addition to its unique developments, it uses almost everything that has already been developed for Snort. So, all Snort roll sets are suitable - Sourcefire VRT, OpenSource Emerging Threats (ETOpen), and commercial Emerging Threats Pro. The output is unified (Unified2), so the result can be analyzed using the usual back ends such as Barnyard2, Snortsnarf, Snorby, Aanval, BASE, FPCGUI, NSM, Sguil and Squert systems. Possible output to PCAP, Syslog, files and the like. For example, Suricata keeps a log of keys and certificates appearing in TLS / SSL connections. In recent releases, Eve log has appeared, which generates event output in JSON format for alerts. The presence of JSON greatly simplifies the integration of Suricata with third-party applications, including systems for monitoring and visualizing logs (like Kibana) [29].

All Suricata and rules settings are made in YAML files. It is more visual and simplifies automatic processing. One of the benefits of Suricata is OSI Layer 7 processing, which greatly enhances its ability to detect malware in various applications. The engine can automatically detect and parse protocols (IP, TCP, UDP, ICMP, HTTP, TLS, FTP, SMB, SMTP, and others). This means that, in the rules, it is not necessary to strictly bind to the port number, as it is done in Snort. It is enough to specify the protocol and action ... Furthermore, the Suricata modules can handle the traffic and detect the protocols, even if a non-standard port is being used.

Rules contain three components: the action (pass, drop, reject, or alert), the title (source and destination IP/port), and the description (what to look for). In the basic configuration, there are only a few rules, and some, of them are even disabled (commented out) in the files themselves. Therefore, it is important to focus on the rules that were connected to Snort using PulledPork. Sometimes, not one, but several TCP connections may be opened between nodes. However, some IDS are not able to see the bigger picture and process each thread separately. Suricata rules make extensive use the concept of flow bits. To track the number of rule triggers, various session variables (for example, using flow int) are used to create counters and flags and then check them. This approach easily copes with trying to guess. In version 2.1, it becomes possible to easily track the rules of the IP source - destination IP (xbits) bundle, which makes it even easier to detect malicious traffic distributed over several connections. In the latest releases, the IP Reputation subsystem has appeared, which can download and utilize data from various databases containing host reputation lists in the rules. A special mechanism enables fast searching and matching with IP addresses. The step-by-step installation process for Snort is detailed in Appendix D. Installing Helper Programs, which is described in Appendix E.

D. Bro (Zeek)

Like Suricata, Zeek (formerly Bro and renamed Zeek at BroCon 2018) is also an intrusion detection system and network security monitoring tool that can detect anomalies such as suspicious or dangerous activity. However, Zeek differs from traditional IDS in that it not only detects exceptions but also captures metadata associated with network activity. This allows for a better understanding of the context of unusual network behavior. This makes it possible, for example, by analyzing an HTTP call or the procedure for the exchange of security

certificates, to look at the protocol, packet headers, and domain names.

If Zeek is considered as a network security tool. We can say that it allows a specialist to investigate an incident, learning about what happened before or during the incident. Zeek also converts network traffic data into high-level events and makes it possible to work with a script interpreter. The interpreter supports a programming language that allows users to interact with events and find out what exactly those events mean in terms of network security. This programming language can also be customized for the interpretation of metadata to suit a particular organization. It allows you to build complex logical conditions using the AND, OR, and NOT operators. This gives users the ability to customize the analysis of their environments. However, it should be noted that, in comparison with Suricata, Zeek may seem to be a rather complicated tool when conducting reconnaissance of security threats [30].

V. CONCLUSIONS

Here we see various approaches to deal with imbalance datasets for intrusion detection. The GAN approach has become very popular today and has shown promised results. Some directions that we want to research with during our current project for rare intrusion detection: Combine various datasets in one – most of them deal with the same sensor data and attack types; Many datasets provide only meta information about networks flow. However, it is interesting also to process entire packet data; Online control of rare class classification performance; Time measurement for various approaches in IDS and their parallelization ability.

ACKNOWLEDGEMENTS

The reported study was funded by VAST (Vietnam Academy of Science and Technology) with project number QTRU01.14/21-22 in 2021.

REFERENCES

[1] Erlacher and F. Dressler, "FIXIDS: A high-speed signature-based flow intrusion detection system," NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, 2018, pp. 1-8. doi: 10.1109/NOMS.2018.8406247.

[2] S. T. Eckmann, G. Vigna, R. A. Kemmerer, “STATL: An attack language for state-based intrusion detection,” in Journal of Computer Security, vol. 10, № 1-2, pp. 71-103, 2002.

[3] S. Roschke, F. Cheng and C. Meinel, "High-quality attack graph-based IDS correlation," in Logic Journal of the IGPL, vol. 21, no. 4, pp. 571-591, Aug. 2013. doi: 10.1093/jigpal/jzs034.

[4] B. Peralta, A. Saavedra and L. Caro, "A proposal for mixture of experts with entropic regularization," 2017 XLIII Latin American Computer Conference (CLEI), Cordoba, 2017, pp. 1-9. doi: 10.1109/CLEI.2017.8226425.

[5] S. Mukkamala, A. H. Sung, A. Abraham, “Intrusion detection using an ensemble of intelligent paradigms,”in Journal of Network and Computer Applications, vol. 28, .№ 2, pp. 167-182, 2005.

[6] I. Ahmad, M. Basheri, M. J. Iqbal and A. Rahim, "Performance Comparison of Support Vector Machine, Random Forest, and Extreme Learning Machine for Intrusion Detection," in IEEE Access, vol. 6, pp. 33789-33795, 2018. doi: 10.1109/ACCESS.2018.2841987.

[7] W. Anani and J. Samarabandu, "Comparison of Recurrent Neural Network Algorithms for Intrusion Detection Based on Predicting Packet Sequences," 2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE), Quebec, QC, Canada, 2018, pp. 1-4. doi: 10.1109/CCECE.2018.8447793.

[8] Dung, N. T., Quân, N. V., & Hùng, N. V. (2023). Application of deep learning model in network reconnaissance attack detection. Journal of Science and Technology on Information Security, 2(16), 60-72.

[9] Tavallaee M, Bagheri E, Lu W, Ghorbani AA (2009) A detailed analysis of the kdd cup 99 data set. In: Proceedings of the Second IEEE international conference on Computational intelligence for security and defense applications, IEEE Press, Piscataway, NJ, USA, CISDA'09, pp 53 – 58.

[10] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue and K. Nakao. Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation, Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security, pp. 29-36, 2011.

[11] P. Gogoi, M. H. Bhuyan, D. K. Bhattacharyya and J. K. Kalita. Packet and flow based network intrusion dataset. International Conference on Contemporary Computing, pp. 322-334, 2012.

[12] Hyunsung Lee, Seong Hoon Jeong and Huy Kang Kim, "OTIDS: A Novel Intrusion Detection System for In-vehicle Network by using Remote Frame", PST (Privacy, Security and Trust) 2017.

[13] Damasevicius, R.; Venckauskas, A.; Grigaliunas, S.; Toldinas, J.; Morkevicius, N.; Aleliunas, T.; Smuikys, P. LITNET-2020: An Annotated Real-World Network Flow Dataset for Network Intrusion Detection. Electronics 2020, 9, 800. https://doi.org/10.3390/electronics9050800.

[14] C. Kolias, G. Kambourakis, A. Stavrou and S. Gritzalis, "Intrusion Detection in 802.11 Networks: Empirical Evaluation of Threats and a Public Dataset," in IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 184-208, Firstquarter 2016, doi: 10.1109/COMST.2015.2402161.

[15] Iman Almomani, Bassam Al-Kasasbeh, Mousa AL-Akhras, "WSN-DS: A Dataset for Intrusion Detection Systems in Wireless Sensor Networks", Journal of Sensors, vol. 2016, Article ID 4731953, 16 pages, 2016. https://doi.org/10.1155/2016/4731953.

[16] Hyunjae Kang, Dong Hyun Ahn, Gyung Min Lee, Jeong Do Yoo, Kyung Ho Park, and Huy Kang Kim, "IoT Network Intrusion Dataset.", http://ocslab.hksecurity.net/Datasets/iot-network-intrusion-dataset, 2019.

[17] Neto, E.C.P.; Dadkhah, S.; Ferreira, R.; Zohourian, A.; Lu, R.; Ghorbani, A.A. CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment. Sensors 2023, 23, 5941. https://doi.org/10.3390/s23135941.

[18] M. Ring. A Survey of Network-based Intrusion Detection Data Sets. arXiv preprint arXiv:1903.02460, 2019.

[19] Pantiukhin D.V, Karelova E. Improving of intrusion classification rate by convolution neural network using training set // Information Technology – 2018. – V. 24. – N 6. – P. 406-413 [in Russian].

[20] Yan, G. Han, and Y. Huang, ``New traffic classification method for imbalanced network data," J. Comput. Appl., vol. 38, no. 1, pp. 20-25, 2018.

[21] K. Jiang, W. Wang, A. Wang and H. Wu, "Network Intrusion Detection Combined Hybrid Sampling With Deep Hierarchical Network," in IEEE Access, vol. 8, pp. 32464-32476, 2020, doi: 10.1109/ACCESS.2020.2973730.

[22] Liu, L., Wang, P., Lin, J., & Liu, L. (2020). Intrusion detection of imbalanced network traffic based on machine learning and deep learning. IEEE Access, 9, 7550-7563.

[23] Vu L., Bui C. T., Nguyen Q. U. A deep learning based method for handling imbalanced problem in network traffic classification //Proceedings of the Eighth International Symposium on Information and Communication Technology. – 2017. – C. 333-339.

[24] Chen, Hongyu, and Li Jiang. "Efficient GAN-based method for cyber-intrusion detection." arXiv preprint arXiv:1904.02426 (2019).

[25] Mari, A.-G.; Zinca, D.; Dobrota, V. Development of a Machine-Learning Intrusion Detection System and Testing of Its Performance Using a Generative Adversarial Network. Sensors 2023, 23, 1315.

[26] Open Source IDS Tools: Comparing Suricata, Snort, Bro (Zeek), Linux, May 2021. [Online]. Available: https://cybersecurity.att.com/blogs/security-essentials/open-source-intrusion-detection-tools-a-quick-overview.

[27] Snoort, October 2005. [Online]. Available: h http://www.thg.ru/network/20051020/index.html.

[28] Base Project, October 2013. [Online]. Available: http://sourceforge.net/projects/secureideas/.

[29] IDS/IPS Suricata, June, 2015. [Online]. Available: https://xakep.ru/2015/06/28/suricata-ids-ips-197/.

[30] Suricata, Snort and Zeek: 3 Open Source Technologies for Securing Modern Networks, 2021. [Online]. Available: https://bricata.com/blog/snort-suricata-bro-ids/.

ABOUT THE AUTHORS

Vu Viet Thang
Workplace: Hanoi University of Industry.
Email: vuvietthang@haui.edu.vn
Education: Received the BSc degree in Computer Science from Mendeleev University of Chemical Technology of Russia in 2008, a MSc degree in Computer Science from Le Quy Don Technical University in 2012, and a PhD degree in Computer Science from Moscow Institute of Physics and Technology in 2019. He is a teacher at faculty of information technology, Hanoi university of industry.
Recent research interests: Clustering; Computer vision; Network security.

Cơ quan làm việc: Đại học Công nghiệp Hà Nội.
Email: vuvietthang@haui.edu.vn
Quá trình đào tạo: Nhận bằng Cử nhân Khoa học máy tính tại Đại học Công nghệ Hóa học Mendeleev - Nga vào năm 2008; Thạc sĩ Khoa học máy tính tại Đại học Kỹ thuật Lê Quý Đôn vào năm 2012; Tiến sĩ Khoa học máy tính tại Viện Vật lý và Công nghệ Moscow - Nga vào năm 2019. Hiện đang là giảng viên khoa Công nghệ thông tin, trường Đại học Công nghiệp Hà Nội.
Hướng nghiên cứu hiện nay: Phân cụm; Thị giác máy tính; An ninh mạng.

Dmitry Valerievich Pantiukhin
Workplace: HSE University, Russia.
Email: dpantiukhin@hse.ru
Education: Ms. Dmitry Pantiukhin is Senior lecturer, National Research University Higher School of Economics (Moscow, Russian Federation) since 2015. He received the master degree on “applied mathematics has physics” at Moscow Institute of Physics and Technology (MIPT), specialization “Neural Networks and Neural Computers” in 2004.
Recent research interests: Neural networks and neural computers; Memristive neuromorphic devices; Cybersecurity and language models.

Cơ quan làm việc: Đại học HSE, Liên bang Nga.
Email: dpantiukhin@hse.ru
Quá trình đào tạo: Nhận bằng Thạc sĩ Toán ứng dụng vật lý tại Viện Vật lý và Công nghệ Moscow - Nga, chuyên ngành Mạng mơ-ron và Máy tính nơ-ron năm 2004.
Hướng nghiên cứu hiện nay: Mạng nơ-ron và máy tính nơ-ron; Thiết bị nơ-ron ghi nhớ; Mô hình ngôn ngữ và an ninh mạng.

Bui Thi Thanh Quyen
Workplace: Institute of Information Technology, Vietnam Academy of Science and Technology.
Email: quyenbt@ioit.ac.vn
Education: Received the BSc degree in Automatic control from Ha Noi University of Technology in 2001, a MSc degree in Automatic control from Ha Noi University of Technology in 2006, and a PhD degree in Intelligent Control and Automation from Pusan National University, Korea in 2013. She is a researcher and a head of Department of Automation Technology, Institute of Information Technology (IOIT), Vietnam Academy of Science and Technology (VAST) since May 2015.
Recent research interests: Embedded systems; Robotics; Measurement systems; Computer vision; Brain-computer interface; IoT.

Cơ quan làm việc: Viện Công nghệ thông tin, Viện Hàn lâm Khoa học và Công nghệ Việt Nam.
Email: quyenbt@ioit.ac.vn
Quá trình đào tạo: Nhận bằng Cử nhân Điều khiển tự động hóa tại Đại học Bách khoa Hà Nội vào năm 2001; Thạc sĩ Điều khiển tự động hóa tại Đại học Bách khoa Hà Nội vào năm 2006; Tiến sĩ về Điều khiển thông minh và Tự động hóa tại Đại học Quốc gia Pusan, Hàn Quốc vào năm 2013.
Hướng nghiên cứu hiện nay: Hệ thống nhúng; Robot; Hệ thống đo lường; Thị giác máy tính; Giao diện não-máy tính; IoT.

Vu Viet Vu
Workplace: CMC University.
Email: vvvu@cmc-u.edu.vn
Education: Received the BSc degree in Computer Science from Ha Noi University of Education in 2000, a MSc degree in Computer Science from Ha Noi University of Technology in 2004, and a PhD degree in Computer Science from Paris 6 University in 2011. He is a researcher at Information Technology Institute, Vietnam National University, Hanoi.
Recent research interests: Clustering; Active learning; Semisupervised clustering; E-governmet applications.

Cơ quan làm việc: Đại học CMC.
Email: vvvu@cmc-u.edu.vn
Quá trình làm việc: Nhận bằng Cử nhân Khoa học máy tính tại Đại học Sư phạm Hà Nội vào năm 2000; Thạc sĩ Khoa học máy tính tại Đại học Bách khoa Hà Nội vào năm 2004; Tiến sĩ Khoa học máy tính tại Đại học Paris 6 vào năm 2011. Hiện đang là nghiên cứu viên tại Viện Công nghệ thông tin, Đại học Quốc gia Hà Nội.
Hướng nghiên cứu hiện nay: Phân cụm; Học tích cực; Phân cụm bán giám sát; Ứng dụng chính phủ điện tử.