# L-IDS: A Lifelong Learning Approach for Intrusion Detection

Hossein Doroud*, Omar Alkhateeb*, Elnaz Alizadeh Jarchlo†, and Falko Dressler*
* School of Electrical Engineering and Computer Science, TU Berlin, Germany
† KPMG AG, Germany
{doroud, alizadeh, dressler}@ccs-labs.org, omar.moh.alkhateeb@gmail.com

***Abstract*—Intrusion detection systems (IDS) represent a class of defensive security tools with the purpose of protecting the network from intruders in the network administrator's arsenal. Despite the high precision of traditional signature-based IDS, its effectiveness is still under question due to the growth of a number of encrypted attacks and the volume of network traffic. This is considered one of the main motivations for researchers to develop anomaly-based IDS, which usually suffer from a higher false positive rate. In this paper, we propose and implement a lifelong-learning anomaly detection IDS (L-IDS) with the capability of the network environment's adaption to limit the false positive rate of anomaly detector in the range of signature-based IDS. We consider Snort as a baseline and UNSW-NB15 as the ground truth in the evaluation of our proposal. We demonstrate how L-IDS achieves a higher level of precision in comparison with the existing signature-based IDS.**

***Index Terms*—Intrusion Detection System; Lifelong Learning; Snort; Deep Packet Inspection; Anomaly Detector**

## I. Introduction

Based on a report by Statista,[1] the number of active internet users worldwide in January 2021 has been increased up to 4.6 billion. This highlights the importance of utilizing a secure digital infrastructure, which protects the end users including companies and individuals from cyber-attacks. Here, intrusion detection systems (IDS) play a significant role to protect computer networks. Two well-known intrusion detection systems are signature-based IDS (S-IDS) and anomaly-based IDS (A-IDS). S-IDS[1] apply deep packet inspection techniques on the incoming traffic and looks for the signatures (often coming from a public database). Whenever the incoming traffic matches one signature, an alarm is raised as an indication suspicious activity. Despite the accuracy of S-IDS, it is resource demanding[2], not applicable over encrypted traffic, and vulnerable against zero-day attacks[1].

In contrast, A-IDS [1] build a network's normal profile considering users' actions and it recognizes activities outside the normal profile as an attack. In this way, A-IDS even process the encrypted traffic and identifies the zero-day attacks. However, any normal changes in the networks affects A-IDS accuracy, drastically. Therefore, there will be a need to update the defined normal profile based on the network changes.

To this end, we propose a Lifelong-learning IDS (L-IDS), which learns during its life time and improves the detections, accordingly. Lifelong-learning[3] is an advanced learning paradigm where the model learns using continuous learning setting. A learner trains from the available data in the training phase and then classifies the incoming data in the testing phase. The learner periodically gets retrained based on incoming data to become more knowledgeable.

Hosseinzadeh et al. [4] highlighted the advantages of suing support vector machine (SVM) as ML algorithm in anomaly-based IDS. However, Read et al. [5] and Losing et al. [6] showed that, despite SVM high performance, it is not applicable in Lifelong-learning schema. In some ML algorithms, the retraining process leads to forgetting the part of knowledge that a model has gained from previous training cycles or the size of new model increase gradually. Therefore, identifying a suitable ML algorithm is essential for developing L-IDS. Recently, the Fuzzy Bounded Twin Support Vector Machine (FBTWSVM) [7] was proposed as an attempt to improve the SVM algorithm. We consider FBTWSVM as an ML algorithm candidate to be used in our proposal.

Our main contributions can be summarized as follows:

- We proposing and implement a novel L-IDS with a mechanism to detect its mistakes and adapt its ML model accordingly;
- we evaluate the FBTWSVM [7] performance as an anomaly detector IDS;
- we compare FBTWSVM with C4.5 [8] and ILVQ [9] ML algorithms from their classification performance and resource consumption perspectives; and
- we evaluate the performance of proposed L-IDS and compare it with `Snort` as a baseline.

## II. Related Work

Choosing a proper dataset for a research study requires substantial analysis and examination. Ring et al. [10] collected several datasets suitable for anomaly detection systems and compared their advantages and disadvantages. We took the dataset of this work and used UNSW-NB15 [11] in our measurements.

Signature-based IDS are known to be highly accurate. They apply string matching to detect any attacks based on their already known signature. `Snort`,[2] `Suricata`,[3] and `Bro`[12] are best known publicly available S-IDS with an active

[1] https://www.statista.com/statistics/617136/digital-population-worldwide/

[2] https://www.snort.org/

[3] https://suricata.io/

community behind. Bhosale and Mane [13] compared these S-IDS from different perspectives such as support platform and resource consumption. The outcome showed that despite the flexibility of `Snort`, it can not perform as good as `Bro` in high-speed networks. As a considerable amont of todays network traffic is using the HTTP protocol, Erlacher and Dressler [14] proposed FIXIDS processing HTTP traffic based on IPFIX features. Their experimental results indicated that FIXIDS was considerably faster than the original `Snort` when analyzing HTTP traffic. Despite the mentioned attempts, researchg on S-IDS is ongoing for processing traffic in high-speed networks. Additionally, all S-IDS are vulnerable to face zero-day attacks due to the inherent delay in preparing and deploying new signatures.

Anomaly-based IDS have been developed as an effort to overcome the mentioned weaknesses. They classify flows outside the normality region as an attack. For example, Ertam et al. [15] compared Naive Bayes (NB), bayer NET(bN), Random Forest(RF), Multilayer Perception, and Sequential Minimal Optimization (SMO) classifying the KDD99 dataset. They showed that RF and bN gave the best results in anomaly detection. Naseer et al. [16] considered various deep neural network structures for anomaly detection models. They compared Deep Convolution Neural Networks (DCNN), Long Short-Term Memory (LSTM), and Convolutional Auto-Encoders with the conventional ML algorithm in anomaly detection domain. Their results indicated that deep learning algorithms perform better than the conventional ML algorithms.

An anomaly detection model requires to be updated after legitimate changes in the network for limiting resulting false positives. Therefore, any highly dynamic ecosystem is a challenging environment for daily operation of anomaly detection-based IDS. To tackle this issue, Noorbehbahani et al. [17] proposed Incremental Semisupervised Flow Network-based IDS (ISF-NIDS), which is an instance-based learning schema. Their evaluation using the KDD99 dataset showed that ISF-NIDS had the capability to operate online and to learn new intrusion efficiently. Later, Constantinides et al. [18] proposed an IDS consist of a Self-Organazing Incremental Neural Network (SOINN) [19] and SVM algorithms. They used the NSL-KDD dataset to measure the performance of their proposal. The experimental results indicated that the proposed anomaly detection system could update and learn online very fast.

Despite the mentioned efforts, we see a gap that we aim bridging by our L-IDS proposal. L-IDS leverages an online labeling system which can be used for reliably updating the ML model. In this paper, we tackle this issue by leveraging the benefits of signature-based IDS.

## III. Lifelong Learning IDS (L-IDS)

Figure 1 illustrates the concept of L-IDS and its main components. First, L-IDS processes the traffic with an anomaly detection component to filter only suspicious or abnormal traffic. The anomaly detector module is an ML-based classifier that identifies the abnormal traffic according to their traffic pattern.

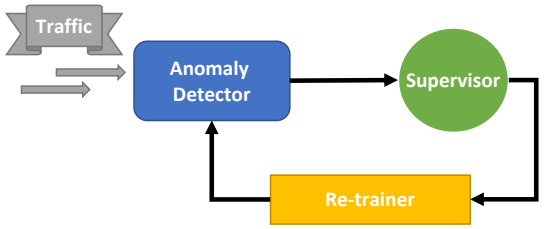


Fig. 1. Conceptual architecture of our L-IDS

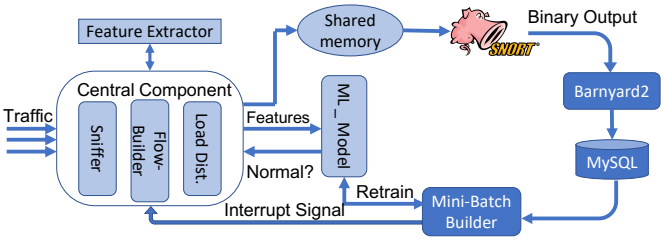


Fig. 2. Components used for implementing our L-IDS

Following, the supervisor re-evaluates the abnormal traffic to identify the false positives of anomaly detection module as well as reporting the details concerning the nature of the detected attacks. The high precision rate of signature-based IDS motivated our design choice to use it as the supervisor. Lastly, the trainer collects the detected false positives from the supervisor and retrains the ML model with the mistakes in a user-defined frequency.

Figure 2 illustrates the realization of the L-IDS system. The central component is the core of the implementation. It captures network packets from the network interface or reads corresponding pcap file and rebuilds the flows with help of Libtrace[4] and libflowmanager,[5] respectively. The central component sends the appropriate information to the feature extractor as well as to the ML model and the supervisor via a shared memory component depending on the state of each flow. The following three states are defined for each flow:

- Feature extraction state: it starts from capturing the first packet of a flow and ends with the arrival of the 5th packet in the flow. The central component forwards each captured packet to the feature extractor for flows that are in this state.
- Anomaly detection state: Upon the reception of the 5th packet of a flow, the central component sends its extracted feature to the ML model.
- Classified state: A flow is in classified state after it is processed by the anomaly detector. The central component either ignores or forwards the new packet of classified flow to the supervisor.

The feature extractor calculates the feature of a flow based on its first five packets that the L-IDS has captured. The central

[4]https://research.wand.net.nz/software/libtrace.php
[5]https://research.wand.net.nz/software/libflowmanager.php

component collects the features and forwards them to ML model. The model classifies the flow as normal or abnormal and announces the result to the central component. The central component sends all the received and upcoming packets which are belonged to the abnormal flows toward the supervisor. The packets of normal flows get dropped from the further investigation.

As the supervisor and the central component are two distinct processes in our implementation, there is a need for a means of communication between the central component supervisor. Linux provides several interprocess communication mechanisms for this purpose. We select the shared memory mechanismas a low delay connection between the central component and the supervisor.

In this work, the well-known signature-based IDS `Snort` is used as a supervisor. It offers different output formats like *fast alert, full alert* and *unified 2*. `Snort` is configured to dump the results in binary format as it adds the minimum overhead to `Snort` process. Barniyard[6] is used to parse the output and save it in a database.

Mini-batch builder reads the false positives from the database and prepares the required data for the re-training procedure. Once the number of detected false positives reaches to the predefine value, it triggers the re-training procedure. Mini-batch builder re-trains a copy of the ML model and then substitutes it with the main model. At the beginning of substitution, the central component receives an interrupt signal to pause sending the feature to the ML model.

## IV. Experimental Setup

### A. Data Sets

The australian Center for Cyber Security published the UNSW-NB15 dataset [11] for supporting research in the filed of IDS. As the original size of UNSW-NB15 dataset (99 GB) is beyond our resource constraints, we consider a subset[7] of it in our experiments. Table I outlines a list of the available attacks and their contribution to the ground truth. The dataset is split up into two parts of training and testing purposes. 20% of the dataset are used for training the anomaly detector and the rest is used for testing the performance of our L-IDS. According to the splitting schema, there is at least one instance from each attack's class in each part.

Despite the capability of supervisor to process the raw traffic, anomaly detectors module (ML) only analyses the characteristics or features of the network traffic. Therefore, defining a set of features that represents the required characteristics of flows for the anomaly detection process is important. In addition, each feature is extractable from the limited information available at the early stage of a flow and without any need to process the application payload with DPI. The former constrain improves the online operation property of the proposal and the latter eliminates adding a barrier to the system for processing the encrypted traffic.

Table II lists the feature set provided by the feature extractor module for the anomaly detector.

TABLE I
Ground truth structure

| Attack Name | Flows | Packets | Size in MByte |
|---|---|---|---|
| Exploits | 409 | 33156 | 28.54 |
| Reconnaissance | 211 | 2354 | 0.28 |
| Fuzzers | 125 | 2067 | 0.7 |
| DOS | 73 | 3062 | 2.058 |
| Generic | 61 | 3908 | 3.55 |
| ShellCode | 25 | 266 | 0.024 |
| Backdoor | 3 | 66 | 0.006 |
| Worms | 2 | 18 | 0.004 |
| Normal | 33793 | 3329686 | 1963.2 |
| **Total** | 34702 | 3374583 | 1998.4 |

TABLE II
List of considered features

| Name | Type | Description |
|---|---|---|
| Service | Categorical | Application layer protocol |
| State | Categorical | State of the flow |
| sbytes | Numeric | Source to destination bytes |
| dbytes | Numeric | Destination to source bytes |
| count | Numeric | Number of flows to the same host as a current flow in the past two seconds |
| srv_count | Numeric | Number of flows to the same service as a current flow in the past two seconds |
| srv_diff_host_rate | Numeric | Percentage of flows to different hosts |
| dst_host_count | Numeric | Count of the flows having same dst host |
| dst_host_srv_count | Numeric | Count of the flows having same dst host and using same service |
| dst_host_diff_srv_rate | Numeric | Percentage of different services on current host |
| src_ttl | Numeric | Source to Destination TTL |
| dst_ttl | Numeric | Destination to Source TTL |
| src_window | Numeric | Source TCP window advertisement |
| dst_window | Numeric | Destination TCP window advertisement |

### B. Supervisor

`Snort` version 2.9.16 has been used with rule-set snortrules-snapshot-29170[8] consisting of 43939 rules. Tjhai et al. [20] demonstrated that utilization of irrelevant rules increases the false positive rate. Therefore, we refine the rule-set by following the trial and error method for minimizing the false positives rate originated from the irrelevant rules. This eliminates a significant number of false positives predictions and increases the precision from only 3.8% to 61.5%. The final rule-set contains 62% of the original one.

[6]https://github.com/firnsy/barnyard2

[7]https://cloudstor.aarnet.edu.au/plus/s/2DhnLGDdEECo4ys/download?path=\%2FUNSW-NB15\%20-\%20pcap\%20files\%2Fpcaps\%2017-2-2015&files=2.pcap

[8]https://www.snort.org/downloads#

### C. Metrics

We measure the performance of the IDS according to accuracy, precision, and recall, which are defined as:

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

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

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

where TP, TN, FP, and FN stand for True Positive, True Negative, False Positive, and False Negative, respectively. In addition, we consider the F1 score to show the trade-off between accuracy and precision:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

All measurements have been conducted on an Ubuntu 20.04.2 platform, which is equipped with an Intel(R) CORE i7 3GHz processor and 16GB memory.

## V. Evaluation and Comparison

An efficient ML algorithm and a representative feature set impact the performance of L-IDS significantly. Therefore, as the first step we evaluate the performance of different ML algorithms to identify the best candidate. In the following, we apply three feature selection algorithms to find the ideal feature set. The evaluation results help us to bias L-IDS in its optimum operation point. In a second step, the performance is compared with `Snort`.

### A. Best ML Algorithm

Read et al. [5] and Losing et al. [6] studied the incremental capability of several ML algorithms. Both reported SVM classifiers showed the highest performance over datasets from different fields. However, the size of SVM model is increasing rapidly after each retraining and is therefore not applicable in a lifelong learning paradigm. They recommended ILVQ [9] and C4.5 [8] as appropriate ML algorithms. Mello et al. [7] proposed FBTWSVM, which was capable of endless learning and had a better performance than SVM. Based on these works, FBTWSVM, ILVQ, and C4.5 are considered as appropriate ML algorithms for the anomaly detector in our proposal.

We trained each ML algorithm with 20% of the ground truth (cf. Table I) and use 80% for the testing/retraining procedure. Table III reports the performance of the classifiers. The C4.5 algorithm shows an average performance based on all the measured parameters. ILVQ does the job with the highest accuracy and FBTWAVM does it with the highest precision, recall and consequently F1. It is mentionable that the accuracy does not represent the performance of a classifier well in an imbalanced dataset [21]. Therefore, we sort Table III in a descending manner based on F1 metric to compare the performance of the ML algorithms.

The speed of traffic processing is an important parameter for the online application of any IDS. We measure the time consumption of different processes related to ML module and the rest of L-IDS processes. This provides us the required information to compare the impact of different ML algorithmu on the final speed of the proposal. Besides, it gives a general impression regarding the speed of L-IDS.

TABLE III
Overall performance of the selected ML algorithms

| Algorithm | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|
| FBTWSVM | 85.83% | **61.99%** | **83.40%** | **71.11%** |
| C4.5 | 85.71% | 38.20% | 30.34% | 50.34% |
| ILVQ | **90.66%** | 14.48% | 16.73% | 26.53% |

L-IDS with each ML algorithm is run for ten times to minimize the effect of any undesirable factors. The considered time intervals are categorised in two groups of online and offline processes. We measured the following time intervals during this experiment:

- Training: The time that is required to train the ML model in an offline mode.
- Retrain: The time needed to re-train each model with the Mini-Batch. It is done in a separate process without adding any delay to the L-IDS detection process operation in its runtime phase.
- ML Classification: The time that is consumed by each ML to classify the whole traffic in the testing dataset.
- Extraction: The time that feature extraction module needs to process the traffic.
- Testing: The time that is consumed by the rest of L-IDS functions (incl. libflowmanager, Snort, SQL, etc) to process the testing dataset.

Figure 3 reports the mean value and standard deviation of L-IDS time consumption configured to use the selected ML algorithms. The figure illustrates the values in two different bars. The green bars are dedicated to the processes that perform

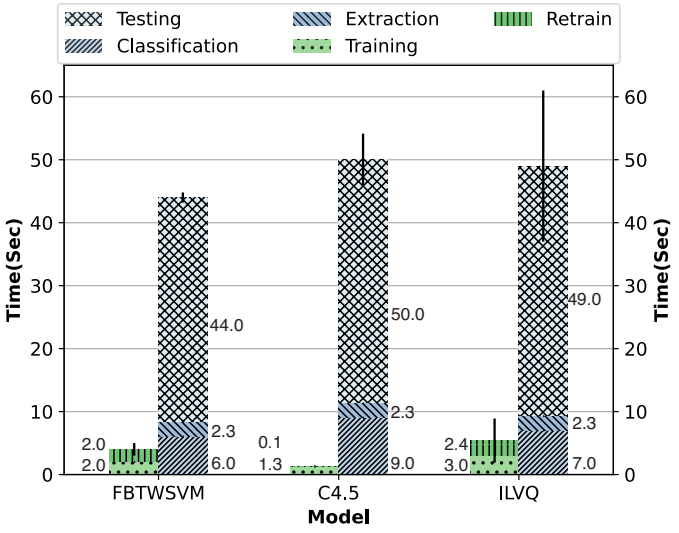


Fig. 3. Time consumption in the respective processing stages

TABLE IV
PERFORMANCE OF L-IDS CLASSIFIERS AND THE SNORT BASELINE SYSTEM

| Metric \ System | Baseline Snort Standalone | L-IDS Anomaly Detector | L-IDS Supervisor |
|---|---|---|---|
| Precision | 61.54% | 61.99% | **96.56%** |
| Recall | 51.91% | **83.40%** | 48.28% |
| F1 score | 56.31% | **71.11%** | 64.37% |

either off-line or in parallel with the main process. Therefore, they do not affect the speed of IDS directly.

The result shows that ILVQ and C4.5 have approximately the same impact on the speed of L-IDS while FBTWSVM has the minimum effect on the proposal speed. Focusing on the training and re-training phase, FBTWSVM shows a middle performance. However, this phase does not influence the processing speed of L-IDS as it can be carry out either in the offline mode, or in parallel with the main process.

Based on the results shown in Table III and Figure 3, has been chosen FBTWSVM as the ML algorithm for anomaly detector module.

### B. Comparison

It is essential to measure the performance of L-IDS and compare it with a well-known IDS for highlighting its strengths and weaknesses. We select `Snort` as the baseline for the comparison. Also, the used supervisor is an identical replica of baseline. The `Snort` instances use the same rule-set in both cases.

Table IV reports the performance of `Snort` and L-IDS for detecting attack flows. L-IDS prosesses the incoming traffic within its anomaly detector and supervisor modules. The anomaly detector labels all the traffic flow according to its normality profile as an attack and the supervisor checks the correctness of the detected attack flows based on its rule-set.

The result show that lifelong learning concept achieves a recall rate up to 83% and limits the false positives rate of the anomaly detector module (ML model) to the range of supervisor in its stand alone version. In addition, it also has a significant positive effect on the supervisor precision by filtering normal flows that are wrongly be matched with the help of a rule of supervisor. However, this filtering decreases 5% the supervisor recall rate in comparison to its stand alone version (Snort stand alone).

We proceed further in this direction to identify the reason behind the decrementation by measuring the recall rate of the classifiers in each attack class. Table V shows that in the majority of classes the anomaly detector has the least false negative and consequently the highest recall rate. Furthermore, a higher recall of anomaly detector makes the recall rate of the supervisor closer to its stand alone version. For example, despite the capability of `Snort` to detect all the *Shellcode* attacks, in the role of supervisor its recall is reduced to the recall rate of anomaly detector module.

L-IDS extends the recall rate of anomaly detector module beyond its supervisor. This can be seen in the *Fuzzers* class.

TABLE V
RECALL RATE OF SNORT STAND-ALONE AND L-IDS (INCL. ANOMALY DETECTOR AND SUPERVISOR) FOR DIFFERENT ATTACK CLASSES

| Attack | Total | Baseline Snort | L-IDS Anomaly Det | L-IDS Supervisor |
|---|---|---|---|---|
| Exploits | 248 | 74.2% | **99.2%** | 72.2% |
| Reconnaissance | 115 | 27.8% | **65.2%** | 23.5% |
| Fuzzers | 68 | 0.0% | **54.4%** | 0.0% |
| DoS | 42 | 57.1% | **90.5%** | 47.6% |
| Generic | 34 | 44.1% | **85.3%** | 44.1% |
| Shellcode | 14 | **100%** | 64.3% | 64.3% |
| Backdoor | 2 | **100%** | **100%** | **100%** |
| Worms | 1 | **100%** | **100%** | **100%** |

The recall rate of `Snort` in this class is 0, which is inline with other research work [22]. *Fuzzers* attacks send random generated input to victims to detect their vulnerability and to exploit it. The highly dynamic nature of *Fuzzers* attack makes it hard to be detected [23]. This motivates [24] to put *Fuzzers* in zero-day attack category that is the weakness of signature-based IDS like `Snort`. However, anomaly detector detects more than 50% of *Fuzzer* attacks by generalising its understanding from the malicious traffic.

## VI. DISCUSSION

Signature-based IDS like `Snort` are well-known for their accuracy. However, their low processing speed makes it challenging to use them in a high speed network. The input packet rate is one of the key factors which affects the load of an IDS. The anomaly detector can reduce the input load by processing all the flows and forwarding only the suspicious ones to the supervisor. The experimental result shows that the input of supervisor is reduced to 9% of packets in this way. Despite this promising result, the current implementation of L-IDS is slower than `Snort` stand alone. L-IDS in the current implementation drops 30% more packets than `Snort` at speed of 50 MByte/s. As a future work, we plan to improve the L-IDS performance in terms of its processing speed.

It is well-known that, generally, anomaly-based IDS suffers from a high false positive rate. Our experiments show that L-IDS improves the precision of the FBTWSVM anomaly detector from 10% to about 62%. As precision improvement is one of the main goals of the L-IDS schema, we compare its performance with `Snort`. However, in future work we also plan to compare the performance of our proposal with different anomaly-based IDSs to study L-IDS performance from other aspects.

Encryption is an effective mechanism that attackers utilise to bypass accurate signature-based IDS like `Snort`. Anomaly detector in L-IDS schema can learn the traffic pattern of any attack from its unencrypted version in re-training cycles and generalizes it to encrypted version of the attack. Preliminary results show that L-IDS detects the attacks which are encrypted by SSL. However, as our dataset contains only two of such attacks, it is not possible to draw a strong conclusion in this regard. Therefore, studying the capability of L-IDS to detect encrypted attacks needs to be further evaluated.

We select FBTWSVM as the ML algorithm after studying previous research works and evaluating its competitors through extensive experiments. However, we observed that the FBTWSVM model size changed from 700 kByte to 1100 kByte after 5 retraining cycles. This limits the feasibility of using FBTWSVM in an L-IDS schema. An appropriate ML algorithm should preserve its performance and its size after each retraining phase. Therefore, developing a ML algorithm appropriate for L-IDS can be another research line for extending this work.

## VII. Conclusion

Despite the rapid evolution of security threats in recent years, the corresponding countermeasures have not been developed at the same pace. To this end, we proposed an IDS with lifelong-learning capabilities (L-IDS). Making use of an anomaly detector and a supervisor module, the system can gradually learn from its false positives. We compared several machine learning (ML) algorithms in terms of accuracy, precision, recall, F1-score, and processing speed to select appropriate candidates for the anomaly detector module. The experimental results showed that FBTWSVM could serve this purpose well. We considered `Snort` as the baseline and compared its performance with L-IDS. The outcome indicated that L-IDS performed better than the baseline and reached to 96% precision and 83% of the recall.

## Acknowledgment

This work has been supported in part by the German Research Foundation (DFG) under grant no. DR 639/20-1.

## References

[1] I. Gondal, A. Khraisat, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, Jul. 2019.

[2] W. Bulajoul, A. James, and M. Pannu, "Network Intrusion Detection Systems in High-Speed Traffic in Computer Networks," in *10th IEEE International Conference on e-Business Engineering (ICEBE 2013)*, Coventry, United Kingdom: IEEE, Sep. 2013, pp. 168–175.

[3] X. Hong, S.-U. Guan, K. L. Man, and P. W. H. Wong, "Lifelong Machine Learning Architecture for Classification," *Symmetry*, vol. 852, May 2020.

[4] M. Hosseinzadeh, A. M. Rahmani, B. Vo, M. Bidaki, M. Masdari, and M. Zangakani, "Improving security using SVM-based anomaly detection: issues and challenges," *Soft Computing*, vol. 25, no. 4, pp. 3195–3223, Oct. 2020.

[5] J. Read, A. Bifet, B. Pfahringer, and G. Holmes, "Batch-Incremental versus Instance-Incremental Learning in Dynamic and Evolving Data," in *11th international conference on Advances in Intelligent Data Analysis (IDA 2012)*, Helsinki, Finland: Springer, Oct. 2012, pp. 313–323.

[6] V. Losing, B. Hammer, and H. Wersing, "Incremental on-line learning: A review and comparison of state of the art algorithms," *Neurocomputing*, vol. 275, pp. 1261–1274, Jan. 2018.

[7] A. R. Mello, M. R. Stemmer, and A. L. Koerich, "Incremental and decremental fuzzy bounded twin support vector machine," *Information Sciences*, vol. 526, pp. 20–38, Jul. 2020.

[8] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[9] Y. Xu, F. Shen, and J. Zhao, "An incremental learning vector quantization algorithm for pattern classification," *Neural Computing and Applications*, vol. 21, no. 6, pp. 1205–1215, Jan. 2011.

[10] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Computers & Security*, vol. 86, pp. 147–167, Sep. 2019.

[11] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Military Communications and Information Systems Conference (MilCIS 2015)*, Canberra, Australia: IEEE, Nov. 2015, pp. 1–6.

[12] R. Sommer, "Bro: An Open Source Network Intrusion Detection System," in *Security, E-Learning, E-Services, 17. DFN-Arbeitstagung überKommunikationsnetze*, J. von Knop, W. Haverkamp, and E. Jessen, Eds., ser. LNI, vol. P-44, Düsseldorf, Germany: GI, Jan. 2003, pp. 273–288.

[13] D. A. Bhosale and V. M. Mane, "Comparative study and analysis of network intrusion detection tools," in *International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT 2015)*, Davangere, India: IEEE, Oct. 2015, pp. 312–315.

[14] F. Erlacher and F. Dressler, "FIXIDS: A High-Speed Signature-based Flow Intrusion Detection System," in *IEEE/IFIP Network Operations and Management Symposium (NOMS 2018)*, Taipei, Taiwan: IEEE, Apr. 2018.

[15] F. Ertam, L. F. Kilincer, and O. Yaman, "Intrusion detection in computer networks via machine learning algorithms," in *International Artificial Intelligence and Data Processing Symposium (IDAP 2017)*, Malatya, Turkey: IEEE, Sep. 2017, pp. 1–4.

[16] S. Naseer et al., "Enhanced Network Anomaly Detection Based on Deep Neural Networks," *IEEE Access*, vol. 6, pp. 48 231–48 246, May 2018.

[17] F. Noorbehbahani, A. Fanian, R. Mousavi, and H. Hasannejad, "An incremental intrusion detection system using a new semi-supervised stream classification method," *International Journal of Communication Systems*, vol. 30, no. 6, Mar. 2017.

[18] C. Constantinides, S. Shiaeles, B. Ghita, and N. Kolokotronis, "A Novel Online Incremental Learning Intrusion Prevention System," in *10th IFIP International Conference on New Technologies, Mobility and Security (NTMS 2019)*, canary Island, Spain: IEEE, Jun. 2019.

[19] F. Shen and O. Hasegawa, "An incremental network for on-line unsupervised classification and topology learning," *Neural Networks*, vol. 19, no. 1, pp. 90–106, Jan. 2006.

[20] G. Tjhai, M. Papadaki, S. Furnell, and N. Clarke, "Investigating the problem of IDS false alarms: An experimental study using Snort," in *IFIP TC 11 23rd International Information Security Conference*, vol. 278, Milano, Italy: Springer, Aug. 2008, pp. 253–267.

[21] S. Al-Azani and E.-S. M. El-Alfy, "Using Word Embedding and Ensemble Learning for Highly Imbalanced Data Sentiment Analysis in Short Arabic Text," *Procedia Computer Science*, pp. 359–366, May 2017.

[22] H. Vargas, C. Lozano-Garzon, G. A. Montoya, and Y. Donoso, "Detection of Security Attacks in Industrial IoT Networks: A Blockchain and Machine Learning Approach," *Electronics*, vol. 10, no. 21, Oct. 2021.

[23] H. N. Thanh and T. V. Lang, "Evaluating Effectiveness of Ensemble Classifiers When detecting Fuzzers attack on the UNSW-NB15 Dataset," *Journal of Computer Science and Cybernetics (JCC)*, vol. 36, no. 2, pp. 173–185, May 2020.

[24] M. Sarhan, S. Layeghy, M. Gallagher, and M. Portmann, "From Zero-Shot Machine Learning to Zero-Day Attack Detection," arXiv, cs.LG, Sep. 2021.