IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 11 January 2024, accepted 4 February 2024, date of publication 19 February 2024, date of current version 23 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3367004

SURVEY

A Systematic Literature Review on Host-Based Intrusion Detection Systems

HAMİ SATILMIŞ[1], SEDAT AKLEYLEK[2,3], AND ZALİHA YÜCE TOK[4]

[1]Department of Computer Engineering, Ondokuz Mayıs University, 55200 Samsun, Turkey
[2]Department of Computer Engineering, İstinye University, 34010 Istanbul, Turkey
[3]Chair of Security and Theoretical Computer Science, University of Tartu, 51009 Tartu, Estonia
[4]ASELSAN, 06200 Ankara, Turkey

Corresponding author: Hami Satilmiş (hami.satilmis@bil.omu.edu.tr)

Zaliha Yüce Tok is supported by the Scientific and Technological Research Council of Turkey (TUBITAK) Frontier R&D Laboratories Support Program 1515 under Grant 5239902.

ABSTRACT With the advancements in computer networks and systems, the number of security vulnerabilities and cyber attacks targeting/using these vulnerabilities continues to increase. Consequently, various intrusion detection systems (IDS) have been developed to detect cyber attacks and ensure information security. IDSs are categorized into two classes based on the data sources: Network-based intrusion detection system (NIDS) and host-based intrusion detection system (HIDS). In this systematic literature review (SLR), studies are examined that focus on HIDS or propose methods applicable to HIDS, as well as those related to IDSs that can be converted into HIDSs. The studies published between 2020 and 2023 are collected from widely used academic databases through various query statements. Filtering based on specific selection and elimination criteria is undergone by the collected studies, resulting in 21 studies for examination. Subsequently, these studies and their advantages and disadvantages are discussed. In addition, while examining the studies, five research questions are addressed. Finally, the defects, potential areas for improvement, and future research directions related to HIDSs are discussed.

INDEX TERMS Intrusion detection system, host-based intrusion detection system, information security, machine learning, deep learning.

I. INTRODUCTION

In parallel with the ongoing advancements in information technology, which continue to progress without slowing down, the number of connected computing devices and the amount of data flow between them rapidly increase. This situation draws the attention of cyber attackers and fuels their appetite. Cyber attacks such as malware and various network attacks aim to compromise the confidentiality, integrity, and availability (CIA) of a computing device or the network it is connected to. Therefore, recognizing and detecting cyber attacks is vital for computer device and network security. Consequently, various intrusion detection systems (IDS) are proposed and developed to recognize and detect cyber attacks. IDS is an information security technology that monitors and analyzes computer networks/systems and detects and reports malicious activities that exploit security vulnerabilities. IDS monitors host/network activities, collects data on the activities and analyzes the collected data to determine if any malicious activity does exist [1], [2]. If there is any attack or suspicious activity, it notifies the system administrator of an alert. IDS are classified into two categories based on the data source: Network-based intrusion detection system (NIDS) and host-based intrusion detection system (HIDS) [3]. Signature-based and anomaly-based detections are two main methods commonly used by IDS [4]. The terms ‘‘attack’’ and ‘‘malicious’’ are used interchangeably throughout the remaining sections of the study.

NIDS detects network-based attacks or suspicious activity by monitoring and analyzing network traffic. NIDS is typically deployed on network devices such as switches

The associate editor coordinating the review of this manuscript and approving it for publication was Yassine Maleh.

and routers to have the ability to monitor the whole network traffic [5]. Most NIDS systems have three stages: monitoring, detection, and response [6]. In the monitoring stage, transmission-related data (the number of packets sent by devices, packet headers, content, size, port numbers, or established connections) is stored [7]. In the detection stage, collected data is processed using signature-based and anomaly-based methods to identify malicious network activities [8]. Signature-based NIDS can detect known attacks with signatures stored in the database. Conversely, anomaly-based NIDS, which usually incorporates statistical, machine learning (ML), and deep learning (DL) methods, can automatically detect unknown and unpredictable attacks [9]. In the response stage, any detected attack or suspicious network activity is reported to the network administrator for appropriate action.

Unlike NIDS, HIDS monitors the single host/device to detect any malicious activity. HIDS systems are typically designed to include functionalities for monitoring, detection, and response to suspicious activities specific to the host device [10]. HIDS primarily monitors process specific activities such as system calls, dynamic link library (DLL) files, system logs, and registry keys and performs analysis on this data to carry out the detection [11]. HIDS are grouped into signature-based and anomaly-based detection methods similar to IDS [12], [13], [14]. While signature-based HIDS detects attacks by matching specific patterns with signatures of known attacks, anomaly-based HIDS uses statistical methods or ML/DL techniques to detect anomalies. In anomaly-based HIDS, ML/DL techniques are trained with normal data samples to create a profile representing normal process activities [5]. Any process activity deviating from this profile is flagged as malicious or suspicious activity by the anomaly-based HIDS, and alerts are sent to the system administrator [8]. Therefore, compared to signature-based HIDS, anomaly-based HIDS systems are more resilient against unknown and zero-day attacks. This SLR is centered on studies primarily focused on or connected to HIDS.

A. RELATED REVIEWS

Review studies published between 2020 and 2023 that directly examine IDS are discussed in this subsection. These review studies are selected from 40 review studies through the study filtering process (detailed in Section III-C). In the final stage, 12 review studies related to IDS are selected. Some of these chosen review studies, like this review, adopted the SLR research method. SLR is a research method that involves selecting studies based on various criteria to examine research studies related to a specific topic and answering predetermined research questions regarding the selected studies. Table 1 provides the characteristics of this SLR and the related review studies.

The CICIDS2018 dataset, which contained many data and attacks commonly used in NIDS, was the focus of the non-systematic review [15]. The issues of overfitting in NIDS that used the CICIDS2018 dataset, the lack of addressing class imbalances in the dataset, and insufficient data cleansing processes were highlighted by the review.

A comprehensive analysis of IDSs proposed for detecting attacks in various network environments was presented in [8]. The management, definition, types, functions, features, and challenges of IDSs were discussed in this review, which adopted the SLR research method.

A thorough description and categorization of IDS types were provided in [4]. Additionally, detection methods, datasets, and feature optimization techniques used in IDS were discussed in the study. A NIDS that achieved a 98.11% accuracy (ACC) rate on the UNSW-NB15 dataset was also suggested.

Multiple aspects of ML/DL methods utilized to detect IDS attacks were systematically examined in [16]. Furthermore, the challenges of ML/DL methods in IDS, datasets used for training and testing these methods, and evaluation metrics were explored in the study.

ML/DL based IDSs were extensively analyzed from various perspectives in [3]. Feature engineering techniques, evaluation metrics, and different application domains of IDS were the focus of this systematic review.

The fundamental features and usage of application-aware HIDS in the automotive industry were researched in [17], a systematic literature review. The architectures, attack detection methods, usage contexts, datasets for training and testing, and performance evaluation metrics of application-aware HIDS were delved into in the study.

The necessity and urgency of an effective real-time anomaly detection system in the internet of things (IoT) environment were emphasized in [2]. In this IoT-focused review, IDS types and attacks in IoT environments were introduced. Additionally, the design requirements for real-time IDS in IoT environments and how an HIDS-based dataset can represent an IoT environment were discussed in the study.

ML/DL methods used in IDS developed for software-defined network (SDN)-based IoT networks were the focus of [18]. Additionally, the use of blockchain technology to improve the security level of SDN-based IoT networks was explored in the study.

IDSs in IoT were systematically examined in [19], and they were classified based on detection methods. Along with the classification of IDSs, details on IDSs suitable for IoT environments, methods used in IDSs, evaluation criteria for IDSs, and popular IDS tools were provided.

In [20], a detailed examination of current research on provenance-based intrusion detection systems (PIDS) was conducted. In this survey, the classification of PIDS was performed, and the need for real-world datasets was discussed. Furthermore, data collection, graph summarization, and intrusion detection in the field of PIDS were addressed.

In [21], a comprehensive evaluation of ML based IDS was conducted. In this review, ML methods and their advantages and disadvantages were extensively examined.

TABLE 1. Comparison of survey/review studies.

Study	Year	Systematic	Type of IDS Focused by Data Source	Dataset Details	IDS Types	Evaluation Metrics	Advantages and disadvantages of ML/DL methods
[15]	2020	No	NIDS	Yes	No	Yes	No
[8]	2021	Yes	NIDS, HIDS	No	Yes	No	No
[4]	2021	No	NIDS	Yes	Yes	Yes	No
[16]	2021	Yes	NIDS	Yes	Yes	Yes	Yes
[3]	2021	Yes	NIDS, HIDS	Yes	Yes	Yes	No
[17]	2021	Yes	HIDS	No	No	No	No
[2]	2022	No	HIDS	Yes	Yes	No	No
[18]	2022	No	NIDS	Yes	Yes	No	No
[19]	2022	Yes	NIDS, HIDS	No	Yes	No	Yes
[20]	2022	No	PIDS	Yes	Yes	No	No
[21]	2023	No	IDS, NIDS	Yes	No	Yes	Yes
[22]	2023	Yes	HIDS	Yes	No	Yes	Yes
This Study	-	Yes	HIDS	Yes	Yes	Yes	Yes

In [22], a systematic review of natural language processing (NLP) use in HIDS was carried out. In this review, NLP methods employed in HIDS were classified. Additionally, datasets and evaluation metrics were mentioned in NLP based HIDS.

B. MOTIVATION AND SCOPE

The studies directly or indirectly related to HIDS are examined in this SLR. Unlike other review studies in Table 1, this study is primarily focused on HIDSs, datasets, IDS types, evaluation criteria, and the advantages/disadvantages of ML/DL methods in HIDSs. The main aim of this SLR is to provide detailed information on recent studies carried out on HIDS. The general scope of this SLR is outlined below:

- The main focus of the survey is determined as HIDS methods and IDS methods which are adoptable to HIDS.
- The studies are collected according to query statements from widely used academic databases such as Scopus, WoS, IEEE Xplore, ScienceDirect, Springer Link, and ACM Digital Library.
- The collected studies are filtered based on predetermined selection and elimination criteria. 21 studies are selected at the final stage.
- During the examination of these 21 studies, efforts are made to answer five research questions.
- Lastly, the limitations of the studies and potential future research directions related to HIDS are discussed.

C. ORGANIZATION

In this study, Section II gives general overview of intrusion detection systems. Section III discusses the systematic review process employed to choose the studies for examination. Section IV summarizes the selected studies. Section V addresses the research questions posed in the study. Section VI highlights the limitations found in the studies and outlines future research directions on HIDS. The final section gives a short overview of SLR. Abbreviations frequently used in this study are given in Table 2.

II. CLASSIFICATIONS OF INTRUSION DETECTION SYSTEMS

IDSs are hardware or software tools that detect and respond to cyber-attacks on networks or host computers [1], [19]. Monitoring and analyzing activities on networks or host computers, identifying anomalous activities, and notifying system administrators are the fundamental functions of IDS [4], [23], [24]. IDSs can be categorized based on data sources, detection methods, and response types. Figure 1 provides a general classification of IDSs.

A. CLASSIFICATIONS OF IDS BY DATA SOURCE

IDSs collect data from networks or host computers to perform the analysis and detect attacks. IDSs are split into two categories based on data sources: NIDSs and HIDSs [2], [3].

1) NETWORK-BASED INTRUSION DETECTION SYSTEMS

NIDSs observe network traffic to detect attacks on the network [3]. Positioned on network devices such as routers

TABLE 2. List of abbreviations.

IDS	Intrusion Detection System	**MI**	Mutual Information
NIDS	Network-based Intrusion Detection System	**LSTM**	Long Short Term Memory
HIDS	Host-based Intrusion Detection System	**ACIDS**	Accurate and Cognitive Intrusion Detection System
SLR	Systematic Literature Review	**DSN**	Destination Sequence Number
ML	Machine Learning	**RREP**	Route Reply
DL	Deep Learning	**TFIDF**	Term Frequencies and Inverse Document Frequencies
DLL	Dynamic Link Library	**SVD**	Singular Value Decomposition
ACC	Accuracy	**NN**	Neural Network
IoT	Internet of Things	**ALAD**	Application-Level Anomaly Detection
SIDS	Signature-based Intrusion Detection System	**TLAD**	Trace-Level Anomaly Detection
AIDS	Anomaly-based Intrusion Detection System	**AUC**	Area Under the Curve
FAR	False Alarm Rate	**TPA**	Temporal Patterns
PIDS	Passive Intrusion Detection System	**TPF**	Temporal Probabilistic Profile
ACTIDS	Active Intrusion Detection System	**BoW**	Bag-of-Word
IDPS	Intrusion Detection and Prevention System	**DMAIDPS**	Distributed Multi-Agent Intrusion Detection and Prevention System
RQ	Research Question	**MOGFIDS**	Multi-Objective Genetic Fuzzy Intrusion Detection System
PRC	Precision	**WE**	Winner Entry
REC	Recall	**DM**	Dolphin Mating
FPR	False Positive Rate	**SEHIDS**	Self Evolving Host-based Intrusion Detection System
DT	Decision Tree	**ANN**	Artificial Neural Network
SVM	Support Vector Machine	**ReNN**	Replicator Neural Network
NB	Naive Bayes	**GRU**	Gated Recurrent Unit
CNN	Convolutional Neural Network	**FCNN**	Fully Connected Neural Network
UGM	Univariate Gaussian Model	**F1**	F1-score
MGM	Multivariate Gaussian Model	**TP**	True Positive
OneR	One Rule	**TN**	True Negative
BN	Bayesian Network	**FP**	False Positive
LR	Logistic Regression	**FN**	False Negative
RF	Random Forest	**PDR**	Packet Delivery Ratio
KNN	K-Nearest Neighbors	**DR**	Detection Rate
HMM	Hidden Markov Model	**TNR**	True Negative Rate
TPR	True Positive Rate	**ROC**	Receiver Operating Characteristic
GBT	Gradient Boosted Trees	**MAE**	Mean Absolute Error
MLP	Multilayer Perceptron	**RMSE**	Root Mean Squared Error
OCSVM	One-Class Support Vector Machine	**RAE**	Relative Absolute Error
IIFS-MC	Improved Infinite Feature Selection for Multi-class Classification	**RRSE**	Root Relative Squared Error
FNR	False Negative Rate	**FSM**	Finite State Machine
CTC	Consolidated Tree Construction	**SRRS**	Supervised Relative Random Sampling
PCA	Principal Component Analysis	**ERT**	Extremely Randomized Trees
IIoT	Industrial Internet of Things	**SKB**	Select K-Best
BPN	Backpropagation Network	**PSO**	Particle Swarm Optimization
ANFIS	Adaptive Neuro-Fuzzy Inference System	**F-GNP**	Fuzzy Graph Neural Process
FCM	Fuzzy C-Mean Clustering	**URL**	Uniform Resource Locators
ENBAG	Ensemble of Bagging Trees	**ENKNN**	Ensemble of K-Nearest Neighbor
ENBOS	Ensemble of Boosted Decision Trees	**ENDSC**	Ensemble of Subspace Discriminator
MRMR	Minimum Redundancy and Maximum Relevance	**CNN-DWA**	CNN and Deep Watershed Auto-Encoder
SWMAT	Sound Wave Memory Analysis Technique	**DTW**	Dynamic Time Warping

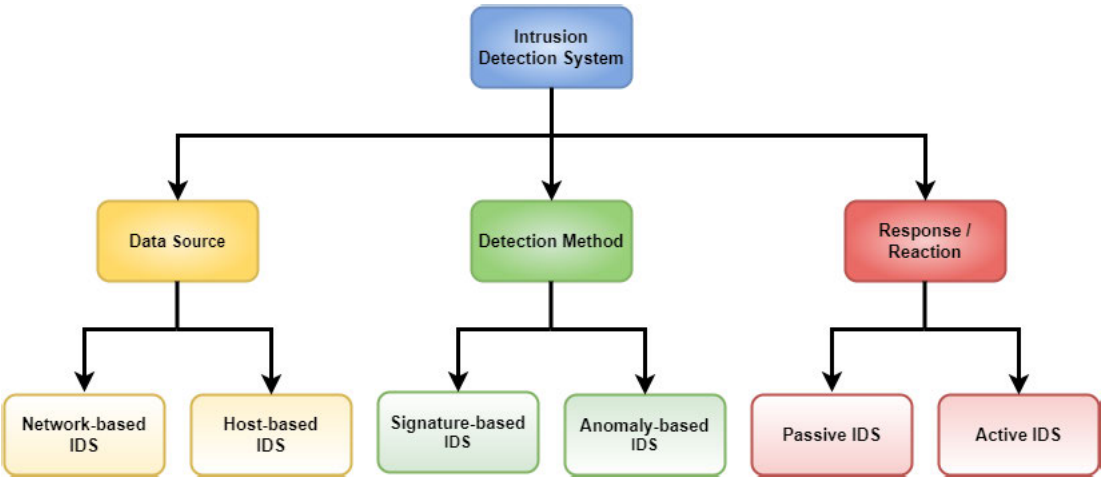

FIGURE 1. Classifications of IDSs.

and switches [5], NIDSs encompass three main functions: monitoring network traffic, detecting attacks, and providing response mechanisms [6]. In NIDSs, sensors are utilized to monitor network traffic and analyze the contents of network packets. Doubtful network packets are captured through these sensors, and based on detection methods, it is determined whether the packets are malicious or not [4]. During the detection stage, network-based data such as the number, headers, contents, size, port numbers of packets, and established connections are used in the detection methods [7]. Network packets identified as attacks by the detection methods are either reported to system administrators through response mechanisms in NIDSs or dropped/blocked.

2) HOST-BASED INTRUSION DETECTION SYSTEMS

HIDSs, monitor the activities of host computers/devices and detect malicious ones among these activities. Positioned on each host computer, HIDSs examine audit trails of the host systems, gather patterns/signatures of activities, and gain insights into the behaviors of these activities [3]. The operational mechanisms of HIDSs generally consist of monitoring various activities on host computers, detecting attacks, and providing response mechanisms [10]. HIDSs that monitor process activities on host computers primarily collect host-specific data such as system calls, DLL files, system logs, and registry keys [11]. Subsequently, these collected data are utilized in the intrusion detection through signature-based or anomaly-based detection methods employed by HIDSs. Suspicious process activities identified by these detection methods are examined for malicious. Malicious activities are either reported to system administrators through response mechanisms in HIDSs or terminated/blocked.

B. CLASSIFICATIONS OF IDS BY DETECTION METHODS

From the perspective of detection methods, IDSs are categorized as Signature-based intrusion detection systems (SIDS) and anomaly-based intrusion detection systems (AIDS) [4], [16], [25].

1) SIGNATURE-BASED INTRUSION DETECTION SYSTEMS

SIDSs, are also called as misuse-based intrusion detection systems, compare activities on networks or host computers with signatures (such as byte codes, text strings, and instruction sequences of known attacks) stored in their databases [26], [27]. When SIDSs find a match between the signatures of known attacks in their databases and the analyzed activities, they classify these activities as attacks and raise alarms. While SIDSs are highly functional and efficient for known attacks, they fail to detect unknown or zero-day attacks [28], [29]. Updating the signature database continuously is proposed as a solution for newly revealed attacks [30].

2) ANOMALY-BASED INTRUSION DETECTION SYSTEMS

AIDSs, also known as profile-based intrusion detection systems, create profiles representing normal activities on networks and host computers. Any activities deviating from these profiles are classified as attacks by AIDSs, which send alerts to system administrators [8]. AIDSs utilize various statistical methods, as well as ML/DL techniques, to construct profiles of normal activities [5], [19]. Due to their ability to detect deviations from normal profiles, AIDSs successfully detect unknown and zero-day attacks [31]. However, a large amount of data is required to create profiles. Moreover, the training time for ML/DL methods used in profile creation can be extensive, leading to high resource consumption. As a result, AIDSs incorporating ML/DL techniques may introduce additional overhead on resource-constrained devices, such as IoT devices [19], [28]. Furthermore, distinguishing boundaries between normal and attack activity profiles can be challenging in AIDSs, potentially resulting in a high false alarm rate (FAR) [32], [33], [34].

C. CLASSIFICATIONS OF IDS BY RESPONSE/REACTION TYPE

When IDSs detect attacks, they respond to these attacks on active or passive manner. In terms of response capabilities, IDSs can be grouped as passive intrusion detection systems (PIDS) or active intrusion detection systems (ACTIDS) [8], [16], [35].

1) PASSIVE INTRUSION DETECTION SYSTEMS

PIDSs observe activities on networks or host computers and detect malicious ones among these activities. However, PIDSs cannot block or prevent the detected malicious activities [3], when faced with malicious activities, they can only respond by alerting system administrators [8].

2) ACTIVE INTRUSION DETECTION SYSTEMS

ACTIDSs respond in real time by blocking or preventing detected attacks on networks or host computers [8]. As a comprehended as intrusion detection and prevention systems (IDPS), ACTIDSs can automatically respond to attacks without requiring intervention from system administrators. On the other hand, ACTIDSs may not be able to conduct an in-depth analysis of activities on networks or host computers [3] and the continuously active response capabilities of ACTIDSs can raise additional overhead on the system.

III. RESEARCH METHODOLOGY

The research methodology applied for selecting and filtering studies related to HIDS between 2020 and 2023 is given in this section. In a more detailed manner, the research questions, query statements used to search for studies in various academic databases, the process of study selection and elimination through filtering, and the changes in the total number of studies are discussed.

A. RESEARCH QUESTIONS

Various aspects of the studies whose main topic is HIDS or related to HIDS are aimed to be explored in this SLR. In line with this objective, the research questions (RQ) and their respective purposes are provided in Table 3. Within the context of the research questions, it is focused on examining the metrics utilized to assess the performance and effectiveness of HIDSs. Additionally, methods employed in HIDSs for detecting activities are investigated, datasets utilized for ML/DL considered, and pros and cons of used ML/DL methods discussed.

B. SEARCH STRATEGY

In this SLR, six academic databases are being scanned to identify research studies related to HIDS. Scopus, WoS (Web of Science), and IEEE Xplore databases provide advanced study search tools with functional search engines, detailed filtering options, and the ability to write query statements. Compared to these databases, ScienceDirect, Springer Link and ACM Digital Library databases contain non-functional search engines and study search tools where query statements cannot be written.

Advanced search tools of Scopus, WoS, and IEEE Xplore databases are used to run query statements. In the case of ScienceDirect, the search is conducted using the title, abstract, and author-keyword fields with the relevant query statement. As for Springer Link and ACM Digital Library databases, the search operations are performed using the respective query statements on the main search engines of these databases.

C. FILTERING PROCESS AND FILTERING CRITERIA

In Table 4, the query statements used for each database are given. To determine which studies will be examined, they need to undergo a filtering process. The selection and elimination criteria utilized for filtering the studies are presented in Table 5.

As the first step of the study filtering process, SC1 and SC2 criteria are applied. Consequently, 226 research or review/survey studies published between 2020 and 2023 are selected. Among these 226 studies, 37 are listed in Scopus, 23 in WoS, 1 in IEEE Xplore, 9 in ScienceDirect, 154 in Springer Link, and 2 in ACM Digital Library. Subsequently, duplicate studies are found and removed from this list, resulting in 191 non-duplicate studies. These non-duplicate studies are categorized as research and review/survey studies using the EC1 criterion in the second filtering step. After the second step, there are 151 research studies and 40 review/survey studies among the non-duplicate ones. The research studies are eliminated based on EC2 criterion and 81 studies published in Q1 and Q2 level journals remain. In the final step of the filtering process, 81 research studies are further evaluated using SC3 and EC3 criteria to determine the studies that will be finally examined. Finally, 23 research studies are identified to be explored. Two studies are identified as irrelevant and excluded from the final set. Therefore, to align with the aim of this SLR, the examination of 21 studies is presented. Figure 2 shows the steps of the studies filtering process and the number of studies remaining after each step.

IV. NOTES ON THE STUDIES

In [36], a HIDS was proposed for detecting malicious activities on Android mobile devices, wherein a combination of statistical and ML methods was utilized. The system log files were analyzed using these statistical methods and ML techniques to identify suspicious activities and calculate the likelihood of their malignancy. Notably, the HIDS was designed to function autonomously on mobile devices, eliminating the need for any server connection. The training and optimization of the HIDS were achieved with only a limited number of malicious data samples, in addition to normal data samples, which proved to be sufficient. During the experimental phase, two real-time datasets were generated using an Android device, and separate experiments were conducted, incorporating both statistical methods and ML techniques. The HIDS

TABLE 3. Research questions and purposes.

	Research Question	Purpose
RQ1	Which metrics are used to evaluate HIDSs?	To learn the metrics utilized for assessing the performance of methods employed by HIDSs in detecting attacks and to comprehend their respective meanings and implications. Examples of commonly used metrics are ACC, precision (PRC), recall (REC), false positive rate (FPR), etc.
RQ2	Which methods are used as detection methods in HIDSs?	To discover the methods used in HIDSs for detecting attacks. Example methods are decision tree (DT), support vector machine (SVM), naive Bayes (NB), convolutional neural network (CNN), etc.
RQ3	Which datasets are used in ML/DL based HIDSs?	To investigate the datasets preferred during the training and testing stages of ML/DL methods used by HIDSs to detect attacks and comprehend the characteristics of these datasets.
RQ4	What are the success rates of the methods?	To examine the best performance rates of ML/DL methods used in HIDS, with which datasets they were obtained.
RQ5	What are the advantages and disadvantages of ML/DL based methods used in HIDSs?	To discuss the advantages and disadvantages of ML/DL based methods used by HIDSs to detect attacks.

TABLE 4. Query statements.

Database	Query Statement
Scopus	TITLE-ABS-KEY("Host-based IDS" OR "Host based IDS" OR "Host-basedIntrusion Detection System" OR "Host based Intrusion DetectionSystem")
WoS	TI =("Host-based IDS" OR "Host based IDS" OR "Host-based Intrusion Detection System" OR "Host based Intrusion Detection System") OR AB =("Host-based IDS" OR "Host based IDS" OR "Host-based Intrusion Detection System" OR "Host based Intrusion Detection System") OR AK =("Host-based IDS" OR "Host based IDS" OR "Host-based Intrusion Detection System" OR "Host based Intrusion Detection System")
IEEE Xplore	(("DocumentTitle": "Host-basedIDS" OR "Host based IDS" OR "Host-based Intrusion Detection System" OR "Host based Intrusion Detection System") OR ("Abstract": "Host-basedIDS" OR "Host based IDS" OR "Host-based Intrusion Detection System" OR "Host based Intrusion Detection System") OR ("Author Keywords": "Host-basedIDS" OR "Host based IDS" OR "Host-based Intrusion Detection System" OR "Host based Intrusion Detection System"))
ScienceDirect	"Host-based IDS" OR "Host based IDS" OR "Host-based Intrusion Detection System" OR "Host based Intrusion Detection System"
Springer Link	("Host-basedIDS" OR "Host based IDS" OR "Host-based Intrusion Detection System" OR "Host based Intrusion Detection System")
ACM Digital Library	Title:("Host-basedIDS" OR "Host based IDS" OR "Host-based Intrusion Detection System" OR "Host based Intrusion Detection System") ORAbstract:("Host-based IDS" OR "Host based IDS" OR"Host-based Intrusion Detection System" OR "Host based IntrusionDetection System") OR Keyword:("Host-based IDS" OR "Hostbased IDS" OR "Host-based Intrusion Detection System" OR"Host based Intrusion Detection System")

featuring either the univariate Gaussian model (UGM) or the multivariate Gaussian model (MGM) as statistical methods demonstrated the highest accuracy, with values ranging from 100% to a minimum of 99.83% accuracy. In contrast, the HIDS employing various ML methods, such as NB, DT, Bayesian network (BN), one rule (OneR), logistic regression (LR), k-nearest neighbors (KNN), random forest (RF), and SVM, achieved an accuracy of 100% for all methods, except for NB.

In [37], a new method based on the hidden Markov model (HMM) was presented for detecting masquerade attacks. This method utilized Unix shell commands as audit data. The approach involved constructing an HMM to represent normal users' behaviors; any behavior that deviated from this HMM was defined as anomalous. In the process of HMM construction, multiple command sequences of different lengths were employed. The HMM was trained and tested using a publicly available Purdue University

TABLE 5. Selection and elimination criteria.

	Selection Criteria		Elimination Criteria
SC1	Studies of research or review/survey type.	**EC1**	Studies review/survey type among research studies.
SC2	Studies published between 2020 and 2023.	**EC2**	Studies published in journals that are not classified as Q1 or Q2 level.
SC3	Studies that utilize ML/DL methods, statistical-based methods for detecting attacks, or propose new algorithms/methods for intrusion detection.	**EC3**	Studies whose main topic is not specifically HIDS or does not relate to general types of IDS, including HIDS.

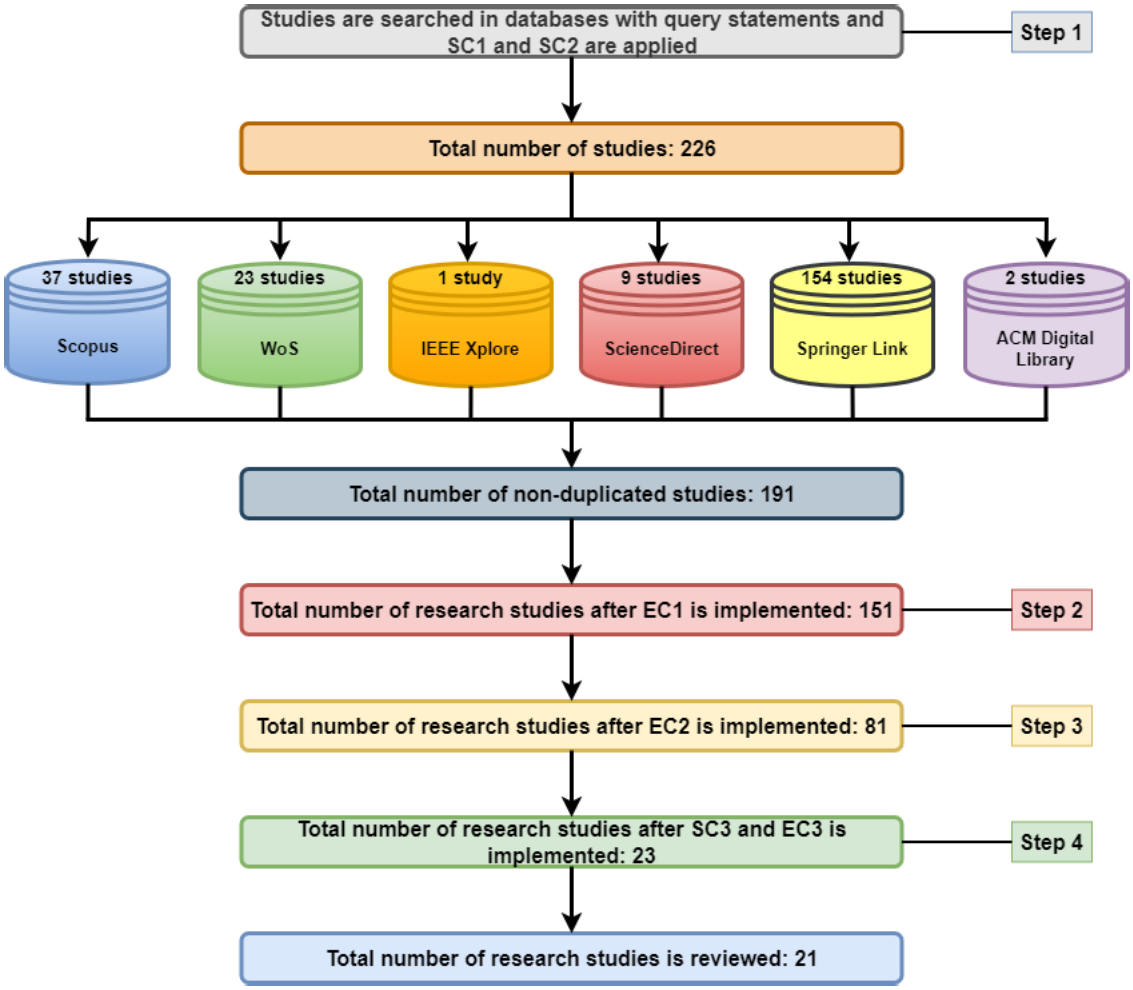

FIGURE 2. The process of studies filtering.

dataset containing Unix shell commands. Notably, during the training process, the parameters of the HMM were computed using parallel computation, which proved to be less computationally costly than the classical Baum-Welch algorithm. The HMM-based method exhibited performance with a FPR of 0.1% and a true positive rate (TPR) of 86.2%. Based on these experimental outcomes, it was figured that the proposed HMM-based method is suitable for online usage with high efficiency.

In [38], an automated HIDS was developed to detect attacks on IoT devices. This HIDS integrated both ML/DL methods and user/kernel data to enhance its effectiveness. The LTTng tracer [39] was employed to collect the devices' traces, represented as system calls. Features were extracted from the collected traces using the Babeltrace API, an open-source code, and numerically represented using a one-hot encoder for the ML/DL methods. The HIDS comprised various ML/DL methods, including DT, RF, gradient boosted trees (GBT), SVM, multilayer perceptron (MLP), and one-class support vector machine (OCSVM). For training and testing, experiments were conducted using a dataset [38] that encompassed data from a home automation system. Assessing the additional overhead introduced by the HIDS, 1.24% overhead on the CPU and 1.87% overhead on memory

observed during a 10-second interval. The experiments revealed that the HIDS achieved the highest accuracy of 100% when using the GBT method, while the SVM method yielded the lowest accuracy of 53.22%. Finally, in terms of predicting the class of activity, the evaluation demonstrated that the DT method required the shortest time of 1.23 ms. In contrast, the GBT method demanded the longest time of 9.28 ms.

In [40], an IDS employing the C4.5 decision tree-based method was proposed. The detection method, J48Consolidated, was based on the consolidated tree construction (CTC) algorithm, enabling efficient operation on datasets with class imbalance. The study also introduced an enhanced version of the supervised relative random sampling (SRRS) sampling method for the preprocessing stage of the detection method, ensuring the creation of a balanced dataset from one with class imbalance. The authors designed and employed a filter-based feature selection and ranking algorithm named improved infinite feature selection for multiclass classification (IIFS-MC) to further improve the IDS's performance. The proposed IDS was developed and evaluated on three datasets: NSL-KDD, ISCXIDS2012, and CICIDS2017. On the NSL-KDD, the IDS achieved impressive performance values of 99.956% ACC and 0.0004% false negative rate (FNR) using all features. On the ISCXIDS2012 dataset, the IDS reached 99.936% ACC and 0.0006% FNR performance with all features. Lastly, when evaluated on the CICIDS2017 with 34 selected features, the IDS demonstrated performance values of 99.955% ACC and 0.0004% FNR.

In [41], a novel context-aware feature extraction method was introduced, aimed at feature selection to decrease classification time and reduce the number of features. An IDS that utilized this feature selection method in its preprocessing stage was proposed, enabling attack types ranging from 4 to 12 to be detected. CNN was also employed as the detection method. To assess the performance of the proposed IDS, the authors conducted various experiments employing the NSL-KDD, CICIDS2017, ADFA-LD, and ADFA-WD datasets. In these experiments, the IDS with the basic CNN method achieved accuracy rates of 81.90%, 99.22%, 95.12%, and 77.01% on the NSL-KDD, CICIDS2017, ADFA-LD, and ADFA-WD datasets, respectively. On the other hand, when the IDS incorporated the CNN method with fine-tuned parameters, it achieved higher ACC rates of 83.43%, 99.29%, and 95.34% on the NSL-KDD, CICIDS2017, and ADFA-LD datasets, respectively. Furthermore, in experiments measuring the classification time per sample, the IDS with the basic CNN method processed samples in 44.00 ms, 87.58 ms, 47.69 ms, and 43.63 ms on the NSL-KDD, CICIDS2017, ADFA-LD, and ADFA-WD datasets, respectively. Additionally, the IDS containing the CNN method with fine-tuned parameters classified samples in 47.71 ms, 87.68 ms, and 49.72 ms on the NSL-KDD, CICIDS2017, and ADFA-LD datasets, respectively.

In [42], three long short term memory (LSTM)-based DL methods were proposed. Among these, two methods utilized either principal component analysis (PCA) or mutual information (MI) to reduce and select the number of features. Consequently, the LSTM-based DL methods were categorized into three distinct approaches: LSTM, LSTM-PCA, and LSTM-MI. The authors conducted several experiments on the KDD99 dataset to assess the performance of the LSTM-based DL methods. In binary classification experiments, the LSTM method achieved an accuracy value of 96.51%, the LSTM-PCA method achieved 99.44% accuracy, and the LSTM-MI method achieved an accuracy of 96.99%. Furthermore, in multi-class classification experiments, the LSTM method obtained a performance rate of 85.65% accuracy, the LSTM-PCA method achieved 99.36% accuracy, and the LSTM-MI method demonstrated accuracy of 96.57%.

In [43], it was highlighted that mobile ad-hoc networks are susceptible to numerous routing attacks due to the lack of central management. To handle this vulnerability and detect attacks on mobile ad-hoc networks, the authors presented a proposed IDS named accurate and cognitive IDS (ACIDS). ACIDS was explicitly designed to detect black hole attacks by identifying the normal behavior of selected parameters. Parameters like destination sequence number (DSN) and route reply (RREP) were utilized by ACIDS to detect potential attacks on the network. ACIDS was evaluated in a mobile network simulation environment known as NS2. Simulation results revealed that ACIDS achieved a higher packet delivery ratio than the AODV routing protocol. However, it was observed that, compared to the AODV protocol without IDS, ACIDS introduced increased end-to-end delay and additional overhead.

In [44], a real-time HIDS capable of detecting anomalous system processes was proposed. The suggested HIDS represented system calls using n-gram feature vector models, and the term frequencies and inverse document frequencies (TFIDF) values of n-gram terms were computed using the TFIDFvectorizer method. To further reduce the dimension of n-gram feature vectors based on TFIDF values, the truncated singular value decomposition (SVD) method was employed. The HIDS included three detection methods: SVM, neural network (NN), and DT. The performance of the HIDS was assessed through experiments using the ADFA-LD and ADFA-WD datasets. In experiments with the ADFA-LD, the HIDS achieved the best binary and multi-class classification performance using the SVM method with a 3-gram feature vector, obtaining FPR values of 3.34% and 9.12%, respectively. On the other hand, experiments on the ADFA-WD demonstrated that the HIDS containing the NN method with a 5-gram feature vector exhibited the best performance. In binary and multi-class classification, this HIDS achieved FPR values of 8.63% and 15.11%, respectively.

In [45], a novel classifier was introduced for anomaly-based HIDS, which represented sequence-to-sequence

behaviors of system calls using DL methods. This classifier, named application-level anomaly detection (ALAD), aimed to discern between normal and anomalous behaviors at the application level. To evaluate the suitability of the ALAD classifier with DL methods, the WaveNet method [46], LSTM method [47], and CNN/RNN method [48] were utilized. ALAD made predictions on anomaly behaviors using models created by DL methods and underwent training and testing with the ADFA-LD and PLAID [45] datasets. Additionally, ALAD was compared with another trace-level anomaly detection (TLAD) classifier. In experiments with the ADFA-LD dataset, ALAD achieved AUC (Area Under the Curve) values of 99.8% with the WaveNet method, 96.6% with the LSTM method, and 98.5% with the CNN/RNN method. For the PLAID dataset, ALAD exhibited AUC values of 99.3% with the WaveNet method, 99.5% with the LSTM method, and 99.3% with the CNN/RNN method. Furthermore, it was observed that ALAD outperformed TLAD in experiments with both datasets.

In [49], a malicious activity detection framework was proposed, which utilized raw multivariate time series data associated with the execution of malicious software. Within this framework, patterns of temporal API calls representing malware behaviors were automatically discovered. Various ML/DL methods were employed for detecting and classifying malware, and these methods were categorized based on data representation types as sequence-based, non-time-oriented, and time-interval-based methods. The sequence-based methods consisted of LSTM and LSTM combined with LR. The non-time-oriented methods included classic techniques such as NB, SVM, LR, and RF. On the other hand, the interval-based methods encompassed temporal patterns (TPA)+NB, TPA+SVM, TPA+LR, TPA+RF, and temporal probabilistic profile (TPF). To evaluate the proposed framework using data samples, a dynamic analysis environment was created using Cuckoo Sandbox [50]. Within this environment, executable malicious files from Windows 10, collected from VirusTotal [51], underwent dynamic analysis to obtain behavior patterns like API calls associated with these files. In the conducted experiments, the TPA+LR method achieved a detection accuracy of 99.4%-99.6% with a FPR of 0.7-0.8% for unknown malware. Moreover, the TPA+SVM method demonstrated the best performance in detecting unknown types of malicious software, with an accuracy of 89.28% and an FPR of 9.69%. Furthermore, in another experiment, the TPA+LR method achieved a classification accuracy of 99.24% for various types of malware.

In [52], it was proposed to preprocess system call patterns used in ML methods for HIDS using bag-of-word (BoW) techniques. Under this proposal, the system call patterns were initially preprocessed using BoW, BoW Boolean, BoW Probability, and BoW TFIDF methods. These preprocessed system call patterns were then used to assess the classification and detection performance of RF, J48 (C4.5), RIPPER, NB, SVM, and KNN methods. The performance of these methods was evaluated on two datasets, namely the ADFA-LD dataset and the VMM malware dataset. In the experiments conducted with the ADFA-LD dataset, the RF method with BoW TFIDF achieved the best performance, attaining 98.4% ACC and 1.7% FAR. On the other hand, in the experiments performed on the VMM dataset, the most successful methods were J48, RF, and RIPPER with BoW TFIDF, achieving 100% accuracy and 0% false alarm rate.

In [53], a new distributed multi-agent intrusion detection and prevention system (DMAIDPS) was introduced to address the challenges of detecting unknown attacks. DMAIDPS employed learning agents to detect anomalous network behaviors. These learning agents applied a set of six stages, including preprocessing, finding similar data, establishing relationship rules, and decision-making processes, to identify anomalous behaviors. The proposed DMAIDPS utilized the K-means method to cluster collected data and employed the KNN method to detect anomalous behaviors based on the clustered data types. DMAIDPS was trained and tested to evaluate its effectiveness using the KDDCup99 and NSL-KDD datasets. In the experimental results, DMAIDPS was compared with several other methods, including DT, NB, 5NN, SVM, multi-objective genetic fuzzy IDS (MOGFIDS) [54], and winner entry (WE) methods. The experiments conducted with the KDDCup99 dataset demonstrated that DMAIDPS achieved improvements of 19.33%, 21.65%, 17.33%, 19.01%, and 14.43% in terms of the ACC metric when compared to DT, NB, 5NN, SVM, MOGFIDS, and WE methods, respectively. On the other hand, experiments conducted with the NSL-KDD dataset revealed that DMAIDPS outperformed DT, NB, 5NN, SVM, MOGFIDS, and WE methods by 17.85%, 16.07%, 10.79%, 14.13%, and 9.94%, respectively, in terms of the ACC metric.

In [55], the dolphin mating (DM) method was proposed to create an IDS with high detection accuracy while requiring less computation time. Inspired by nature, the DM method was designed to select features from datasets effectively. Its effectiveness was evaluated through experiments that measured the performance of various ML methods, including NN, DT, KNN, and Bagging. The experiments utilized the NSL-KDD and Kyoto datasets. When using the NSL-KDD, the Bagging method, which incorporated the 13 features selected by the DM method, achieved the finest performance with an accuracy rate of 99.79%. Similarly, for the Kyoto dataset, the Bagging method, which utilized the eight features selected by the DM method, attained the highest accuracy rate of 99.84%. On the other hand, in experiments where feature selection was not applied, the ML methods achieved lower success rates than experiments with feature selection.

In [56], a self evolving host-based intrusion detection system (SEHIDS) was proposed to detect attacks on IoT networks. The main idea behind SEHIDS was to incorporate artificial neural network (ANN) methods that could self-train and update their architectures online whenever their performance decreased on each IoT node. This approach ensured

that the nodes contained ANN methods capable of detecting attacks against them with unique architectures. SEHIDS was categorized into two common types of IDS: SIDS and AIDS. The SIDS type of SEHIDS utilized the MLP method as the detection technique, while the AIDS type employed the replicator neural network (ReNN) method to detect attacks. To evaluate SEHIDS, it was tested on the BoT-IoT, IoTID20, and TON-IoT datasets. According to the experimental results, SEHIDS' SIDS type detected various attacks on all three datasets, achieving a minimum TPR of 99% and a maximum TPR of 100%. On the other hand, SEHIDS' AIDS type achieved performance rates of 100% ACC on the BoT-IoT and IoTID20 datasets and 99.9% ACC on the TON-IoT dataset. Regarding resource consumption, the experiments indicated that SEHIDS imposed low overhead on IoT devices.

In [5], a stacking ensemble-based HIDS was developed as a security mechanism against attacks on Windows operating systems. The HIDS analyzed system files containing DLL instruction calls performed by diverse applications and system processes to detect anomalous system calls. DL techniques, including LSTM, gated recurrent unit (GRU), Bi-LSTM, Bi-GRU, and fully connected neural network (FCNN), were utilized as the detection methods. The system process files containing sequences of DLL calls underwent preprocessing using n-gram, Word2Vec, and Glove methods. The Word2Vec method generated numerical vector representations and inputs to LSTM methods. On the other hand, the Glove method provided numerical vector representations used as inputs to GRU methods. The FCNN method combined the outputs of the LSTM and GRU methods to make the final decision. ADFA-WD dataset is chosen for performance evaluation. In binary-class experiments on the ADFA-WD, HIDS with a 7-gram term size achieved 91.1% ACC with a 100-dimensional vector and 11.5% FPR with a 150-dimensional vector. In multi-class experiments on the same dataset, HIDS with a 7-gram term size and a 100-dimensional vector attained the highest performance with 68.7% ACC and 7.1% FPR.

In [57], an IDS incorporating the backpropagation network (BPN) method was developed. Particle swarm optimization (PSO) was employed for optimizing BPN, and feature extraction was performed in the preprocessing phase with PSO. The BPN method was evaluated using system calls from the KDDCup99 dataset. The performance of BPN was compared with adaptive neuro-fuzzy inference system (ANFIS), fuzzy graph neural process (F-GNP), and fuzzy c-mean clustering methods. BPN achieved a performance of 96.5% DR and outperformed other methods in terms of the DR metric.

In [58], an ML based detection system was developed to identify malicious uniform resource locators (URLs). The detection system had two tasks: first, classifying URLs as benign or malicious using a binary classifier, and second, predicting the classes of URLs. Four ensemble learning methods, the ensemble of bagging trees (ENBAG), the ensemble of k-nearest neighbor (ENKNN), the ensemble of boosted decision trees (ENBOS), and the ensemble of subspace discriminator (ENDSC) were employed in the developed detection system. Features for these methods were selected using the minimum redundancy and maximum relevance (MRMR) algorithm. The performance of the detection system was measured using the ISCX-URL2016 dataset. The ENBAG method emerged as the most successful model, with ACC values of 99.3% and 97.9% for binary and multi-class classification, respectively.

In [59], a hybrid method was proposed to overcome feature selection and imbalanced data challenges in IDSs. The method, called Convolution neural network and deep watershed auto-encoder (CNN-DWA), was trained and tested using the KDDCup99 dataset. Label encoding was used in the preprocessing stage to convert categorical variables in KDDCup99 to numerical values. The performance of CNN-DWA was evaluated, and it demonstrated an ACC of 98.05%, outperforming CNN with an ACC of 94.54%.

In [60], concerns were raised about the coverage and timeliness of existing datasets related to HIDSs. To address this issue, a large-scale dataset named DongTing was created to detect anomalies in Linux kernels. DongTing, comprising 85 GB of raw data and containing 18,966 system calls categorized as normal or abnormal, was compared with other datasets. The comparison was conducted on methods such as CNN/RNN [48], LSTM [47], WaveNet [46], and ECOD [63]. Methods trained with DongTing achieved the best generalization scores, and those trained with abnormal data showed better generalization ability than those trained with normal data.

In [61], a technique was proposed to create fingerprints for IoT devices by transforming dynamic memory traces into sound wave signals using a lossless transformation function. This technique, called the sound wave memory analysis technique (SWMAT), extracted mel-frequency cepstral coefficients (MFCC) during the transformation process. Short-time Fourier transform (STFT) was employed to extract MFCCs from sound waves. The SWMAT technique utilized dynamic time warping (DTW) distance measure to score the similarity between two MFCC sequences. SWMAT was evaluated using 125 MFCC sequences from 20 benign and five infected IoT applications. In the evaluation, SWMAT had a performance accuracy of 95%.

In [62], a novel intrusion detection framework was proposed for analyzing system call sequences to detect known or unknown attacks. A hybrid model incorporating the LSTM method and a system call frequency-based anomaly detection technique was composed the proposed framework. Attacks were classified by the LSTM using information from previously analyzed system call sequences. The classes of attacks were predicted by the frequency-based anomaly detection technique, including BoW, n-gram, and TFIDF, which are preprocessing methods. The final decision on

TABLE 6. The contributions of the studies.

Study	Main Contribution
[36]	A HIDS was developed to autonomously detect suspicious behaviors on Android devices without requiring any server connection. Various methods, including OneR, DT, NB, BN, LR, RF, KNN, SVM, UGM, and MGM, were employed in the HIDS to detect attacks.
[37]	A novel method based on HMM was proposed for detecting masquerade attacks on networks. The parameters of HMM are computed in parallel to reduce costs.
[38]	A host-based framework incorporating ML/DL methods was proposed for automatically detecting attacks on IoT devices. User/kernel data were employed in the framework's DT, RF, GBT, SVM, OCSVM, and MLP methods.
[40]	An IDS using the C4.5 decision tree-based method as a detector was developed. The SRRS sampling method and IIFS-MC feature selection algorithm were also proposed to balance class-imbalanced datasets.
[41]	A context-aware feature extraction method was proposed as a preprocessing step for multi-class intrusion detection. The suggested method was employed in a CNN-based IDS.
[42]	LSTM methods using PCA and MI methods were applied to detect attacks.
[43]	ACIDS was designed to detect black-hole attacks on mobile ad-hoc networks. Parameters like DSN and RREP were utilized by ACIDS to detect potential attacks.
[44]	A novel HIDS, incorporating n-gram, TFIDFvectorizer, and SVD methods, was proposed to detect anomalous system processes. SVM, NN, and DT methods were employed as detectors within the HIDS.
[45]	A new classifier for anomaly-based HIDS was introduced, which represented the sequence-to-sequence behaviors of system calls using DL methods. This classifier, named ALAD, was implemented with WaveNet, LSTM, and CNN/RNN methods.
[49]	A detection framework was designed to capture the malware behaviors by analyzing raw multivariate time series data using various ML/DL methods. The ML/DL methods in the framework, utilizing API calls, were categorized into non-time-oriented, sequence-based, and time-interval-based methods based on data representation types.
[52]	The combined use of BoW methods and ML techniques in HIDSs for attack detection was proposed. System call patterns processed with BoW methods in the preprocessing stage were employed in RF, J48, RIPPER, NB, SVM, and KNN methods for detecting attacks.
[53]	DMAIDPS was designed for the purpose of detecting and preventing unknown attacks. In DMAIDPS, the K-means and KNN methods were employed in conjunction.
[55]	For IDSs to detect attacks more efficiently, a feature selection method (DM) was proposed to select optimum features in data sets. The effectiveness of this method was evaluated on NN, DT, KNN, and Bagging methods.
[56]	A new HIDS, incorporating self-evolving and updating ANN methods, was proposed to detect IoT network attacks. Named SEHIDS, this HIDS was developed to include signature-based detection by default, incorporating the MLP method. In addition, SEHIDS, with the ReNN method, was utilized as an anomaly-based IDS.
[5]	A stacking ensemble-based HIDS utilizing DL methods was proposed to detect anomalous processes on Windows operating systems. In this HIDS, system process files were underwent preprocessing using n-gram, Word2Vec, and Glove methods. LSTM, GRU, and FCNN methods were used as detection methods.
[57]	An IDS incorporating the BPN method was proposed. The PSO was employed to optimize BPN and in the feature extraction process.
[58]	A detection system was developed using ensemble learning methods to detect malicious URLs and predict their classes.
[59]	CNN-DWA was introduced to address feature selection and imbalanced data challenges in IDSs.
[60]	A large-scale dataset called DongTing was created to be used in detecting anomalies in Linux kernels.
[61]	A method called SWMAT was proposed to transform dynamic memory traces in IoT devices into sound wave signals, extract MFCCs from these signals, and find similarities between MFCC sequences.
[62]	An intrusion detection framework that incorporates the LSTM method and system call frequency-based anomaly detection technique and analyzes system call sequences was developed.

the types of system call sequences was determined by the weighted average ensemble method. The framework's performance was evaluated on the ADFA-LD dataset, achieving an accuracy of 97.2% and a FPR of 2.4%.

Table 6 briefly outlines the contributions of the studies. Table 7 summarizes the advantages and disadvantages of the studies. Additionally, the advantages and disadvantages mentioned in Table 7, a common advantage among the studies [36], [38], [45], [49], [60], and [61] is that new datasets were created. On the other hand, the common disadvantage of studies other than [38] and [61] is that experiments were not carried out in real-world environments in addition to laboratory experiments.

V. ANSWERS TO RESEARCH QUESTIONS

In this section, research questions are addressed to examine the studies from various perspectives.

A. WHICH METRICS ARE USED TO EVALUATE HIDSs?

Various metrics were utilized to evaluate the performance of the HIDSs, IDSs, or methods in the examined studies. The definitions, mathematical equations, and preferred studies of these metrics are given in Table 8. Upon reviewing Table 8, it is observed that ACC, PRC, REC, F1-score (F1), and FPR metrics are commonly used for evaluating ML/DL methods. Accordingly, to calculate these types of metrics, TP, FP, TN, and FN are defined as follows.

TABLE 7. The advantages and disadvantages of the studies.

Study	Advantages	Disadvantages
[36]	-Detected suspicious behaviors autonomously without the need for any server connection. -Adjustment of ML and statistical methods detected attacks with only sufficient normal data samples. -Detected suspicious behaviors with high ACC rates.	-There is no detail on feature selection methods, representation of features, and data preprocessing stages.
[37]	-Detected online masquerade attacks. -Suitability of the proposed method for use as detectors in HIDSs. -Performed with low FPR values. -Potential efficiency in terms of computational cost.	-While it performed better regarding the TPR metric compared to similar studies in the literature, the TPR value was still low. -Although it was claimed to be efficient regarding computational cost, no experiments were conducted to substantiate this claim.
[38]	-Detected anomalies with high ACC rates. -Evaluated the proposed framework with real threats in a home automation system.	-The success rates of SVM and MLP methods were meager. -ML/DL methods that perform well could detect similar attacks, but their performance in different attack types remains unknown. -The absence of feature selection methods and the lacked evaluation of ML/DL methods with different feature variations.
[40]	-Performed with high ACC and low FNR values. -Outperformed other similar proposed IDSs in the literature.	-Being a standalone signature-based HIDS that could be combined with an anomaly detection engine to enhance the detection rate.
[41]	-Detected attacks with high performance on the CICIDS2017 and ADFA-LD datasets. -Conducted experiments with both host-based and network-based data samples.	-Low success rate on the NSL-KDD and ADFA-WD datasets. -The achieved ACC rate using the CICIDS2017 dataset was lower than some of the proposed IDSs in the literature.
[42]	-LSTM methods using the PCA technique exhibited high binary and multi-class classification performance. -Compared with similar proposed methods in the literature, the REC rates were high. -The two-feature LSTM method using PCA was suited to large-scale and high-dimensional spaces.	-LSTM methods using the MI technique showed low performance in multi-class classification. -Compared with similar proposed methods in the literature, the ACC rates were low.
[43]	-Increased the packet delivery ratio of the AODV protocol. -Improved the routing discovery performance of the AODV. -Tested in a mobile network simulation environment.	-Increased the routing overhead in the network nodes. -Caused a growth in the number of control packets sent and received during transmission. -Increased end-to-end delay.
[44]	-Detected anomalies with high ACC rates and low computational overhead. -Outperformed other similar proposed HIDSs in the literature. -Potential suitability for real-time usage.	-In experiments involving multi-classification, the FPR rates were high. -The computation of TFIDF values for n-gram terms produced extra overhead.
[45]	-The ALAD classifier demonstrated higher ACC and lower FPR values than the TLAD classifier. -ALAD incurred almost no additional overhead compared to TLAD.	-In experiments conducted with the ADFA-LD dataset, certain LSTM and CNN/RNN methods performed less than their counterparts.

TABLE 7. *(Continued.)* **The advantages and disadvantages of the studies.**

Study	Advantages	Disadvantages
[49]	-Detected unknown malware with high ACC and low FPR values. -Interval-based methods achieved the most successful performances.	-Detected unknown types of malware with low ACC and high FPR values. -Excluded of malicious software samples that can hide their malicious intent and activities within a virtual machine (able to sense the virtual machine) in the experiments. -The possibility of high-time consumption makes it difficult to implement in real-time applications.
[52]	-Generally, detected attacks with high ACC rates. -In experiments with the VMM malicious software dataset, some methods achieved 100% ACC and 0% FAR performance.	-In experiments with the VMM malware dataset, methods not using BoW TFIDF reached high FAR rates. -Performance values were lower in experiments conducted with the ADFA-LD dataset compared to the VMM malware dataset.
[53]	-Generally achieved better ACC and REC metrics performance than NB, DT, SVM, 5NN, MOGFIDS, and WE methods.	-In experiments on the NSL-KDD dataset, low ACC rates for normal and U2L and PRB attack type data samples. -Detected of data samples belonging to the R2L attack type in the KDDCup99 dataset with lower ACC, REC, and F1 values than the NB method.
[55]	-Methods using DM had higher ACC rates than those not using DM. -The methods using DM were more successful in detecting attacks than those proposed in similar studies in the literature.	-DM-using and non-DM-using ML methods had low ACC rates in experiments where the NSL-KDD was employed as training and testing datasets.
[56]	-Signature-based or anomaly-based SEHIDS detected attacks with high ACC rates. -Anomaly-based SEHIDS outperformed other similar proposed HIDSs in the literature. -ANN methods could improve and update their architectures according to different attacks. -Potential suitability for use in IoT devices in terms of resource consumption.	-The signature-based SEHIDS was not compared to similar proposed HIDSs in the literature.
[5]	-Outperformed other similar proposed HIDSs in the literaturez regarding ACC rates.	-Low ACC rates in multi-classification experiments. -High FPR values as a result of binary and multi-classification experiments.
[57]	-Compared to other methods, the DR value was higher.	-Compared to other methods, the FAR value was higher.
[58]	-ENBAG and ENKNN methods classified URLs with high ACC values. -ENBAG, ENKNN, and ENBOS methods successfully detected malicious URLs with high ACC values. -Achieved better ACC values than the methods in other similar studies.	-ENBOS and ENDSC methods classified URLs with low ACC values. -ENDSC method detected malicious URLs with a low ACC value.
[59]	-Achieved high ACC values. -The CNN-DWA method outperformed the CNN method in terms of performance.	-Performance values were not compared with similar methods in the literature.
[60]	-The DongTing dataset elevated the performance of DL methods compared to other datasets. -DL methods trained with anomalous data outperformed those trained with normal data.	-The number of abnormal system call sequences was limited. -Abnormal system call sequences were not labeled into subclass types.

TABLE 7. *(Continued.)* **The advantages and disadvantages of the studies.**

Study	Advantages	Disadvantages
[61]	-It was demonstrated that memory images can be effectively utilized to create fingerprints.	-The computation time for DTW was long.
[62]	-Achieved high ACC values. -Better performance values in terms of the ACC metric were observed compared to similar studies in the literature.	-In terms of the FPR metric, better values were obtained in a similar study in the literature. -The difficulty in detecting attack types that can alter their behaviors.

- TP: Number of correctly predicted positive data samples.
- FP: Number of incorrectly predicted negative data samples.
- TN: Number of correctly estimated samples of negative data.
- FN: Number of incorrectly predicted positive data samples.

In addition to the metrics provided in Table 8, the studies also employed specific evaluation metrics tailored to their respective subjects. In [38], the additional overhead on CPU and memory created by the proposed HIDS was utilized to assess its performance. Furthermore, in [56], the suggested HIDS was evaluated based on memory usage and the number of floating-point operations per second (FLOPS). The performance of the IDS designed to detect attacks, as described in [43], was measured using parameters such as packet delivery ratio (PDR), routing overhead, packet loss (bytes), throughput (kbps), and end-to-end delay (ms). The loss performance metric, which measures the distance between the predictions of a method and the actual values, was used in [59] and [62]. The computation time of DTW distance was employed as an evaluation metric in [61].

B. WHICH METHODS ARE USED AS DETECTION METHODS IN HIDSs?

In studies, various methods were employed for detecting and classifying attacks. The majority of these methods were constituted by ML/DL techniques. Table 9 provides an overview of the detection methods utilized in these studies. Table 9 also information regarding preprocessing methods and datasets.

In [36], various detection methods, such as OneR, DT, NB, BN, RF, KNN, and SVM, were utilized. In addition to these ML methods, statistical methods like UGM and MGM were employed for detecting attacks. The experimental results demonstrated that these methods can detect attacks with high ACC rates.

In [37], attacks were detected using the HMM method. In the experiments, it was claimed that the HMM method, which was performed with low FPR values, can be efficient regarding computational cost. The HMM method also had low TPR values as a disadvantage.

In [38], FSM and one-hot encoder methods were employed during the preprocessing stage. DT, RF, GBT, SVM, MLP, and OCSVM methods were preferred for the attack detection phase. All methods except SVM and MLP achieved high ACC rates in the experiments. Among these methods, the DT method emerged as the one that predicted attack classes in the least amount of time. On the other hand, the GBT method, which had the highest ACC rate, appeared to consume the highest time while predicting attack classes.

In [40], the data were preprocessed with the SRRS sampling method and the IIFS-MC feature selection and ranking algorithm. A C4.5-based method relying on the CTC algorithm was employed to perform the task of detecting attacks. This detection method, J48Consolidated, demonstrated high ACC and low FNR values, indicating its strong performance in attack detection.

In [41], the feature selection process was performed using the ERT and SKB methods. A DL method, CNN, was utilized for detecting attacks. In experiments on the CICIDS2017 and ADFA-LD datasets, the CNN method achieved high ACC rates in detecting attacks. However, the CNN method showed low ACC rates in NSL-KDD and ADFA-WD datasets experiments. Moreover, the CNN method exhibited a long per-sample classification time.

In [42], the PCA and MI methods were employed for feature selection and dimensionality reduction in the preprocessing stage. The LSTM methods, with or without these preprocessing methods, were used to detect attacks. These detection methods, categorized as LSTM, LSTM-PCA, and LSTM-MI, demonstrated their performance in detecting attacks. LSTM-PCA, among these methods, achieved high ACC rates in binary and multi-class classification experiments. Additionally, it was found that LSTM-PCA is suitable for large-scale and high-dimensional domains. The LSTM-MI method, another detection method, detected attacks with higher ACC rates than the LSTM method, which does not include size reduction and feature selection methods.

In [43], an ACIDS algorithm was applied to detect attacks. This algorithm utilized the parameters DSN and RREP, and based on a predefined threshold value, it identified attacks. According to the experiments, the ACIDS algorithm significantly improved packet delivery and efficiency rates compared to the AODV protocol. It also contributed to a reduction in packet loss. On the other hand, end-to- end delay time cause routing overhead.

In [44], SVD dimensionality reduction, n-gram and TFIDFvectorizer methods were used in the pre-processing

TABLE 8. The evaluation metrics used in studies.

Metric	Definition	Equation	Studies in Which It was Used
ACC	The rate of correctly predicted data samples.	$\frac{TP+TN}{TP+TN+FP+FN}$	[36], [38], [40], [41], [42], [49], [52], [53], [55], [56], [5], [58], [59], [61], [62]
PRC	The rate of correctly predicted positive data samples to total positively predicted data samples.	$\frac{TP}{TP+FP}$	[36], [38], [41], [42], [44], [52], [53], [55], [56], [5], [58], [59], [62]
REC or sensitivity or TPR or detection rate (DR)	The rate of correctly predicted data samples that are positive.	$\frac{TP}{TP+FN}$	[36], [37], [38], [40], [41], [42], [44], [45], [49], [52], [53], [55], [56], [5], [57], [58], [59], [62]
F1 or F-measure	The harmonic mean of PRC and REC metrics.	$2 \times \frac{PRC \times REC}{PRC+REC}$	[36], [38], [41], [42], [44], [52], [53], [55], [56], [5], [58], [59], [60], [62]
FPR or FAR	The rate of positive predicted negative data samples to total negative data samples.	$\frac{FP}{FP+TN}$	[37], [40], [44], [45], [49], [52], [55], [5], [57], [60], [62]
FNR	The rate of falsely predicted negative data samples to total positive data samples.	$\frac{FN}{FN+TP}$	[40], [55]
True negative rate (TNR)	The rate of correctly predicted negative data samples to total negative data samples.	$\frac{TN}{TN+FP}$	[55], [59]
Misclassification Rate	The rate of incorrectly predicted data samples.	$\frac{FP+FN}{TP+TN+FP+FN}$	[40]
Receiver operating characteristic (ROC) curve	The curve shows the relationship between FPR and TPR.	-	[37], [45], [55]
AUC	The area under the ROC curve. It measures the predict performance of the methods.	-	[45], [55], [60]
Analysis Latency	The average time it takes a method to predict the class of a data sample.	-	[38]
Testing Time/Instance or Classification Time/Sample or Detection Time or Inference Time	The average time consumed when predicting the class of a data sample.	-	[40], [41], [45], [56], [58], [60]
Training Time	The time consumed from the training phase of ML/DL methods.	-	[40], [45], [56]
Mean absolute error (MAE)	The absolute mean difference between the values predicted by a method and the correct values.	$(\frac{1}{n}) \times \sum \lvert true\ value - predicted\ value \rvert$	[40]
Root mean squared error (RMSE)	The root mean square value of the absolute difference between the values predicted by a method and the true values.	$\sqrt{(\frac{1}{n}) \times \sum \lvert true\ value - predicted\ value \rvert^2}$	[40]
Relative absolute error (RAE)	The rate of the absolute difference between the values predicted by a method and the true values to the absolute difference between the true values.	$\frac{\sum \lvert true\ value - predicted\ value \rvert}{\sum \lvert true\ value - average\ true\ values \rvert}$	[40]
Root relative squared error (RRSE)	The rate of the square of the absolute differences between the true values and the values predicted by a method and the square of the absolute differences between the true values.	$\sqrt{\frac{\sum \lvert true\ value - predicted\ value \rvert^2}{\sum \lvert true\ value - average\ true\ values \rvert^2}}$	[40]

stage. SVM, NN, and DT methods were used for the attack detection phase. The experiments revealed that anomalies were detected with low computational load. Due to this low computational load, it can be inferred that this study's preprocessing and detection methods were suitable for real-time usage. However, in experiments involving multi-class classification, high FPR values were observed. Additionally, the computation of TFIDF values for n-gram terms introduced an extra load for dimensionality reduction.

In [45], ALAD and TLAD classifiers were used as detectors, which collected the predictions of the WaveNet, LSTM, and CNN/RNN methods. In the experiments, the ALAD classifier demonstrated high ACC rates. Compared to the TLAD classifier, the ALAD classifier introduced almost no additional overhead. However, it was noted that DL methods may not be suitable for real-time applications due to their long training and high detection times.

In [49], various ML/DL methods were utilized as detectors, categorized into three: non-temporal, sequence-based, and time-interval-based, according to their data representations. The detection methods consisted of non-temporal methods such as NB, SVM, LR, and RF, sequence-based methods like LSTM and LSTM+LR, and time-interval-based methods including TPA+NB, TPA+SVM, TPA+RF, and TPF. When examining the experimental results, it was observed that unknown malware was detected with high ACC and low FPR values. However, unknown types of malware were detected with low ACC and high FPR values. Moreover, due to the possibility of high time consumption, it was suggested that these methods might be challenging to use in real-time applications. Conversely, according to the experimental results, the most successful methods were the time-interval-based ones.

In [52], the BoW, BoW boolean, BoW probability, and BoW TFIDF methods were used during the preprocessing stage. ML methods, including RF, J48, RIPPER, SVM, KNN, and NB, were employed to detect attacks. Throughout the experiments, attacks were generally detected with high ACC rates. The best performance values were achieved with the J48, RF, and RIPPER methods incorporating the BoW TFIDF method in experiments using the VMM malware dataset, reaching 100% ACC and 0% FAR values. On the other hand, experiments on the VMM malware dataset showed that detection methods without the BoW TFIDF method often had high FAR values, while most detection methods operated with low FAR values when ADFA-LD dataset used.

In [53], attacks were detected by an approach consisting of K-means, a clustering method, and KNN, a classification method. The experiments showed that the approach performed well regarding ACC and REC metrics compared to NB, DT, SVM, 5NN, MOGFIDS, and WE methods. However, in experiments on the NSL-KDD, it was noticed that normal and U2L and PRB attack types data samples were detected with low ACC rates.

In [55], a method called DM was utilized for feature selection during preprocessing. The selected features were used by the ML methods, namely NN, DT, KNN, and Bagging, to detect attacks. In the experiments, ML methods employed the DM method outperformed those not using it, achieving higher ACC rates. Additionally, the Bagging method reached high ACC values in experiments using the NSL-KDD training and Kyoto datasets. However, experiments where the NSL-KDD was selected as the training and test datasets had low ACC rates.

In [56], DL methods, namely MLP and ReNN, were used for detecting attacks. In the experiments, both methods successfully detected attacks with high ACC values. Moreover, it was mentioned that in terms of resource consumption, MLP and ReNN methods had the potential to be efficient.

In [5], the preprocessing stage consisted of Word2Vec and Glove word embedding and n-gram methods. In the first phase of the detection process, LSTM and Bi-LSTM methods used the Word2Vec method, and GRU and Bi-GRU methods used the Glove method were utilized. In the final detection phase, the FCNN method combined the outputs of LSTM and GRU methods to detect attacks. As a result of the experiments, it was noticed that higher ACC rates are achieved compared to the ACC rates in similar studies in the literature. In contrast, low rates of ACC were observed in experiments on multiple classifications. In addition, high FPR values were achieved in binary and multiple classification experiments.

In [57], BPN, which uses features extracted in the preprocessing phase with PSO, was used as the detection method. BPN achieved high DR values and outperformed similar methods in terms of DR metric. On the other hand, BPN had lower performance than other methods in the FAR metric.

In [58], malicious URLs were detected using ensemble learning methods, namely ENBAG, ENKNN, ENBOS, and ENDSC. Among these methods using features selected with the MRMR algorithm, ENBAG emerged as the most successful in binary and multi-class classification.

In [59], the CNN-DWA method, developed to overcome feature selection and imbalanced data challenges, was employed for attack detection. Using label encoding in the preprocessing stage, CNN-DWA outperformed the CNN method, achieving a high ACC value.

In [60], CNN/RNN, LSTM, WaveNet, and ECOD methods were employed to evaluate the DongTing dataset. Among these methods detecting anomalous system call sequences in DongTing, CNN/RNN and WaveNet stood out in terms of performance compared to other methods.

In [61], similarities between MFCC sequences were determined by the SWMAT method using DTW distances. In experiments, SWMAT performed with high ACC values.

In [62], a hybrid intrusion detection framework incorporating the LSTM method and a system call frequency-based anomaly detection technique was used to detect attacks. The anomaly-based detection technique of this framework consisted of NN and RF methods. In the preprocessing stage of this detection technique, BoW, n-gram, and TFIDF were used. The framework achieved high ACC and low FPR values.

C. WHICH DATASETS ARE USED IN ML/DL BASED HIDSs?

The training and testing stages of ML/DL based methods were undergone with various datasets. These datasets were chosen based on the types and characteristics of

TABLE 9. An overview of the detection methods.

Study	Detection Methods	Preprocessing Methods	Datasets
[36]	ML methods: OneR, DT, NB, BN, LR, RF, KNN, and SVM. Statistical methods: UGM and MGM.	No detailed information.	Two real datasets created within the scope of the study.
[37]	HMM.	There were no other details given other than the data cleaning and setting operation.	A publicly shared dataset from Purdue University.
[38]	DT, RF, GBT, SVM, MLP, and OCSVM.	LTTng monitor. Babeltrace API feature extractor. Finite state machine (FSM). One-hot encoder.	A dataset containing data from a home automation system created as part of the study.
[40]	J48Consolidated.	SRRS sampling method. IIFS-MC feature selection and ranking algorithm.	NSL-KDD. ISCXIDS2012. CICIDS2017.
[41]	CNN.	Extremely randomized trees (ERT). Select k-best (SKB).	NSL-KDD. CICIDS2017. ADFA-LD. ADFA-WD.
[42]	LSTM, LSTM-PCA, and LSTM-MI.	PCA. MI.	KDD99.
[43]	ACIDS algorithm.	No information.	No dataset.
[44]	SVM, NN, and DT.	n-grams. TFIDFvectorizer. SVD dimension reduction method.	ADFA-LD. ADFA-WD.
[45]	WaveNet. LSTM. CNN/RNN. TLAD and ALAD.	No information.	ADFA-LD. A dataset called PLAID created as part of the study.
[49]	Non-temporal methods: NB, SVM, LR, and RF. Sequence-based methods: STM and LSTM+LR. Time-interval-based methods: TPA+NB, TPA+SVM, TPA+LR, TPA+RF, and TPF.	Information gain.	Within the scope of the study, a dataset created using Cuckoo Sandbox.
[52]	RF, J48, RIPPER, SVM, KNN, and NB.	BoW, BoW boolean, BoW probability, and BoW TFIDF.	ADFA-LD. VMM malware dataset.
[53]	K-means. KNN.	There were no other details provided other than the normalization and feature type convert process.	KDDCup99. NSL-KDD.
[55]	NN, DT, KNN, and Bagging.	DM feature selection method.	NSL-KDD. Kyoto.
[56]	MLP. ReNN.	No information.	BoT-IoT. TON-IoT. IoTID20.
[5]	LSTM and Bi-LSTM with Word2Vec. GRU and Bi-GRU with Glove. FCNN combining the outputs of LSTM and GRU methods.	n-grams. Word2Vec word embedding method. Glove word embedding method.	ADFA-WD.
[57]	BPN.	PSO.	KDDCup99.
[58]	ENBAG, ENKNN, ENBOS, and ENDSC.	MRMR.	ISCX-URL2016.
[59]	CNN-DWA.	Label encoding.	KDDCup99.
[60]	CNN/RNN. LSTM. WaveNet. ECOD.	No information.	DongTing. ADFA-LD. PLAID.
[61]	SWMAT.	No information.	The dataset created within the scope of the study.
[62]	LSTM and NN+RF.	BoW, n-grams, and TFIDF.	ADFA-LD.

the data samples. For instance, the ADFA-LD dataset was used in ML/DL based HIDS due to its inclusion of information related to system calls in the Linux operating system [41], [44], [45], [52]. Table 10 summarizes the features of datasets utilized in the studies. Except for datasets created explicitly for the studies in Table 10, commonly used and accessible datasets, such as ADFA-LD, ADFA-WD, and PLAID are generally utilized in HIDSs. In contrast, most other widely used datasets are usually preferred in NIDSs. Brief description of datasets are given below.

- In [36], two real-time datasets were created using an Android device, containing five different types of malware and normal samples. It was observed that these nameless datasets, which contain an equal number of benign and malicious data samples, contribute positively to the performance of ML methods in the experiments in [33]. Because the distribution of data samples is an important factor that directly affects the performance of ML methods.
- The dataset publicly shared by Purdue University [64] includes shell command tokens obtained from Unix users' command histories. This dataset is widely used in the field of anomaly-based masquerade detection. Therefore, the results of studies using this dataset can be compared with each other [65].
- In [38], a dataset was created containing data from a home automation system. This nameless dataset includes 58% benign and 42% attack data samples. As a result, this dataset, which can be considered a balanced dataset, is naturally predicted to cause less bias than imbalanced datasets.
- The KDD99 or KDDCup99 dataset [66] contains sample data related to normal and attacks network traffic on computer networks. In KDDCup99, a legacy dataset, the number of attack data samples is considerably more significant than the normal data sample. Hence, this dataset is imbalanced and negatively affects ML methods' performance. In addition, raw data samples must be processed before this dataset can be used.
- The advanced version of the KDDCup99, known as the NSL-KDD [67], excludes redundant data samples present in the KDDCup99 dataset. In this dataset, the numbers of normal and attack data samples are close to each other. However, the number of normal data samples is considerably higher than the number of data samples belonging to the four different attack types. As a result, NSL-KDD is an imbalanced dataset. Additionally, this dataset contains different types of feature values and conversion is needed to improve the performance of ML methods.
- The ISCXIDS2012 [68] and CICIDS2017 [69] datasets consist of attack and normal data samples related to computer network traffic. Since the proportion of normal data samples in ISCXIDS2012 is 97.2%, this dataset has a high-class imbalance. For ML methods to reach high-performance values, this imbalance problem needs to be solved. As a solution, data sampling is commonly performed to increase the number of attack data types. On the other hand, the CICIDS2017 dataset contains recent attacks and normal data samples. This dataset has missing values and class labels. Therefore, data with missing values and class labels need to be removed before using this dataset. Also, this dataset's number of normal data samples is considerably higher than the attack data samples. This problem, which creates a class imbalance, needs to be resolved so that it does not adversely affect the performance of ML methods.
- The ADFA-LD [70] and ADFA-WD datasets [71] contain system call traces used by hosts under normal and attack conditions. Each sample in these datasets consists of a trace. The ADFA-LD includes system call traces from the Linux operating system, while the ADFA-WD contains DLL call traces from the Windows operating system. In addition, ADFA datasets, which are based on HIDS, focus on real attack scenarios and system vulnerabilities. The data samples in these datasets contain only categorical data types.
- The PLAID dataset introduced in [45] covers modern system calls and up-to-date attack data samples, considering the deficiencies of the ADFA-LD dataset. The characteristics of this dataset were not provided in the mentioned study; hence, they are omitted in Table 10.
- The nameless dataset created in [49] contains executable files from the VirusTotal malware repository collected on the Windows 10 operating system.
- The VMM malware dataset [72] includes malware samples gathered from the Zoo repository and virus samples accumulated from the VxHeaven repository. In this dataset, the number of malware data samples is more than 26 times that of normal data samples. Therefore, there is a high-class imbalance in this dataset.
- The Kyoto2016+ dataset [73] consists of samples related to computer network traffic. Fourteen fundamental characteristics were derived from the KDDCup99 dataset. This dataset contains numeric and non-numeric conditional properties. The pre-processing stage is needed before use.
- The BoT-IoT [74] and IoTID20 [75] datasets contain various attack and normal data samples related to IoT networks. The number of attack data samples in both datasets is much higher than the normal data samples. This situation causes the bias problem on ML methods.
- The TON-IoT dataset [76], created by the developers of the BoT-IoT, aims to supply a more comprehensive dataset containing various attack and normal data samples threatening industrial IoT (IIoT). Before using this dataset, property value type conversion and missing values need to be taken care of. In addition, the class imbalance problem must be resolved.
- The ISCX-URL2016 [77] dataset comprises examples of both benign and malicious URLs. The number of

malicious URL examples in this dataset is approximately twice that of benign ones. Consequently, when developing ML/DL methods with this dataset, addressing the issue of imbalanced data is necessary.
- The DongTing dataset [60] is as the first dataset specifically designed for detecting anomalies in Linux kernels. This dataset includes system call sequences categorized as normal and abnormal. The number of abnormal system call sequences in this dataset is greater than that of the normal ones, resulting in an imbalance issue.
- The dataset created in [61] consists of 125 MFCC sequences obtained from 20 benign and five infected IoT applications. This dataset is the first dataset encompassing memory traces of both normal and infected IoT applications.

D. WHAT ARE THE SUCCESS RATES OF THE METHODS?

The detection methods used in the studies were tested by various experiments. In the experiments, the performances of detection methods were evaluated, especially with different datasets and feature combinations. Summary of the experimental results of the detection methods is given in Table 11. Number of features and datasets used on these experiments are also added to table.

Two real-time datasets were generated using an Android device in [36]. SVM and RF methods showed the best performance among ML methods with 100% ACC when 13 or 15 features selected. Among the statistical methods, the best performance values were reached by UGM with 100% ACC when 15 features were selected.

In [37], Purdue University's dataset was used when evaluating the performance of the HMM method. In experiments, four users' shell command streams were preferred. The highest performance values in the results of the experiments were 0.1% FPR and 86.2% TPR.

In [38], the dataset created in the study was used to measure the success of ML/DL methods in detecting attacks. In experiments with 13 features, an overhead of 1.24% on CPU and 1.87% on memory occurred in 10 seconds. In anomaly detection, GBT showed the best performance with 100% ACC. In terms of prediction time, DT was the most successful method, with 1.23 ms.

In [40], the J48Consolidated method used as the detection method and evaluated on the NSL-KDD, ISCXIDS2012, and CICIDS2017 datasets. In the experiments performed with the NLS-KDD, 99.962% ACC and 0.0004 FNR values were observed when all features were used. In the ISCXIDS2012, 99.936% ACC and 0.0006 FNR values were reached with all features. The method had 99.955% ACC and 0.0004 FNR when 34 features were selected on CICIDS2017 dataset.

In [41], the success of the CNN method was measured on the NSL-KDD, CICIDS2017, ADFA-LD, and ADFA-WD datasets. In experiments in the NSL-KDD, fine-tuned CNN achieved 83.43% ACC with 30 features. In CICIDS2017 dataset, fine-tuned CNN performed with 99.29% ACC when 57 features selected. In the ADFA-LD, the fine-tuned CNN with the feature vector of size 350 had an ACC of 95.34%. In the ADFA-WD, the baseline CNN reached 77.01% ACC.

In [42], the KDD99 was used to evaluate the attack detection performance of different combinations of the LSTM method. In the evaluations, it was observed that the LSTM-PCA method had the best performance in binary and multi-classification. With two features, LSTM-PCA performed with 99.44% ACC in binary classification and 99.36% ACC in multi-classification.

In [43], when evaluating the ACIDS algorithm, it was compared with the AODV protocol. As a result of the comparison, it was observed that the ACIDS algorithm increased the PDR by at least 40%. Additionally, the number of lost packets was less.

In [44], the performance of ML methods was evaluated on ADFA-LD and ADFA-WD datasets. On ADFA-LD dataset, using a 3-gram feature vector, SVM achieved a 3.34% FPR in binary classification, and SVM performed with a 9.12% FPR by using 3 gram feature vector in multi-class classification. On the ADFA-WD, NN with a 5-gram feature vector reached an 8.63% FPR in binary classification, and NN performed a 15.11% FPR in multi-class classification.

In [45], ADFA-LD and PLAID datasets were used. The combination of ALAD and WaveNet methods in the ADFA-LD had an AUC of 99.8%. The combination of ALAD and LSTM methods performed with 99.5% AUC on PLAID dataset.

In [49], the success of detection methods was measured by the dataset created within the scope of the study. The TPA+LR method was the most successful for detecting unknown malware and classifying malware types. This method worked in unknown malware detection experiments with 99.6% ACC and 0.7% FPR. In addition, it had 99.24% ACC in the classification of malware. On the other hand, the TPA+SVM method achieved the best performance values in detecting unknown types of malware with 89.28% ACC and 9.69% FPR.

In [52], ML methods used as detectors were evaluated with ADFA-LD and VMM datasets. In the experiments on the ADFA-LD, where the BoW TFIDF method was used in the preprocessing stage, RF performed with 98.4% ACC and 1.7% FAR. In the VMM, J48, RF, and RIPPER methods reached 100% ACC and 0% FAR, with the BoW TFIDF method.

In [53], the performance of DMAIDPS was compared with NB, DT, SVM, 5NN, MOGFIDS, and WE methods. KDDCup99 and NSL-KDD were used as datasets for comparison. On the KDDCup99, DMAIDPS outperformed at least 14.43% for ACC and at least 12.3% for REC. On the NSL-KDD, there was an improvement of at least 9.94% in terms of ACC and at least 11.07% in terms of REC.

In [55], the success of ML methods in detecting attacks was evaluated in experiments with NSL-KDD and Kyoto datasets. In the NSL-KDD, DT performed best with 83.539% ACC when 13 features were used. In experiments where

TABLE 10. An overview of datasets.

Dataset	Attack Data Samples and Numbers	Normal Data Samples and Numbers	Ratio (Attack/ Normal)	Total Number of Data Samples	Number of Features	Studies in Which It was Used
The two datasets in [36].	Samples of five different types of malware: 12000	Normal data samples: 12000	1.00	24000	15	[36]
Dataset of Purdue University.	-	-	-	66450 shell command tokens.	-	[37]
The dataset in [38].	42% of the entire number of data samples.	58% of the entire number of data samples.	0.72	-	13	[38]
NSL-KDD	DoS: 53385, U2R: 252 R2L: 3749, Probing: 14077 Total: 71463	Normal data samples: 77054	0.92	148517	41	[40] [41] [53] [55]
ISCXIDS2012	2.8% of the entire number of data samples.	97.2% of the entire number of data samples.	0.02	2450324	20	[40]
CICIDS2017	19.7% of the entire number of data samples.	80.3% of the entire number of data samples.	0.24	2830743	80	[40] [41]
ADFA-LD	Samples of six different types of attacks: 746 traces.	Normal data samples: 833 traces. Normal validation data samples: 4373 traces.	0.89	5952 traces.	-	[41] [44] [45] [52] [60] [62]
ADFA-WD	Samples of 12 different types of attacks: 5773 traces.	Normal data samples: 356 traces. Normal validation data samples: 1828 traces.	16.21	7957traces.	-	[41] [44] [5]
KDD99 or KDDCup99	DoS: 3883349, U2R: 52 R2L: 208, Probing: 41102 Total: 3924711	Normal data samples: 972781	4.03	4897492	53	[42] [53] [57] [59]
The dataset in [49].	Malware samples: 11453	Normal data samples: 5779	1.98	17232	-	[49]
VMM malware dataset.	Attack data samples: 954 traces.	Normal data samples: 36 traces.	26.50	990 traces.	-	[52]
Kyoto2016+	Attack data samples: 88915	Normal data samples: 74837	1.18	163752	24	[55]

TABLE 10. *(Continued.)* An overview of datasets.

Dataset	Attack Data Samples and Numbers	Normal Data Samples and Numbers	Ratio (Attack/ Normal)	Total Number of Data Samples	Number of Features	Studies in Which It was Used
BoT-IoT	Information gathering: 1821639 DoS: 71537674 Information theft: 1587 Total: 73360900	UDP: 7225 TCP: 1750 ARP: 468 IPV6-ICMP: 88 ICMP: 9 IGMP: 2 RARP: 1 Total: 9543	7687.40	73370443	32	[56]
TON-IoT	Backdoor: 508116, DoS: 9540336 Injection: 452659, MITM: 1052 Password: 1718568 Ransomware: 72805 Scanning: 7140161 XSS: 2108944 Total: 21542641	Normal data samples: 796380	27.05	22339021	46	[56]
IoTID20	DoS: 59391 Mirai: 415677 MITM: 35377 Scan: 75265 Total: 585710	Normal data samples: 40073	14.61	625783	86	[56]
ISCX-URL2016	Spam: Around 12000 Phishing: Around 10000 Malware: Over 11500 Defacement: Over 45450 Total: Around 79000	Normal data samples: Over 35300	2.24	Around 115000.	79	[58]
PLAID	Attack data samples: 1145 traces.	Normal data samples: 39672 traces.	0.03	40817 traces.	-	[45] [60]
DongTing	Attack data samples: 12116 traces.	Normal data samples: 6850 traces.	1.77	18966 traces.	-	[60]
The dataset in [61].	Attack data samples: 25 MFCC sequences.	Normal data samples: 100 MFCC sequences.	0.25	125 MFCC sequences.	-	[61]

NSL-KDD and Kyoto were used together, the Bagging method reached 99.833% ACC with eight features.

In [56], BoT-IoT, IoTID20, and TON-IoT datasets were used in experiments to measure the performance of SEHIDS. SEHIDS reached 100% ACC with low resource consumption and computational costs in all three datasets.

In [5], the combination of LSTM, GRU, and FCNN methods was evaluated with the ADFA-WD. As a result of the evaluations, 91.1% ACC was reached when binary classification, 7-gram term size, and 100-dimensional vector were used. A 68.7% ACC value was observed in multi-classification when the same term size and vector dimensional were used.

In [57], the KDDCup99 dataset was used to evaluate the BPN method, where PSO was used in the preprocessing stage. BPN achieved a DR of 96.5% and a FAR of 4.4% in experiments. Although BPN demonstrated the best performance in the DR metric compared to ANFIS, F-GNP, and FCM methods, it had the highest FAR values.

In [58], the performance of the ENBAG, ENKNN, ENBOS, and ENDSC methods was measured using the ISCX-URL2016 dataset. These ML methods were evaluated for binary and multi-class classification, utilizing 59 features selected by the MRMR algorithm. ENBAG emerged as the most successful method in both classification scenarios, achieving ACC values of 99.3% and 97.9%, respectively.

In [59], the CNN-DWA method, employing label encoding in the preprocessing stage, was trained and tested on the KDDCup99 dataset. Compared to the CNN, this method demonstrated the most successful performance with an ACC rate of 98.05%.

In [60], CNN/RNN, LSTM, WaveNet, and ECOD methods were evaluated using the DongTing, ADFA-LD, and PLAID datasets. In the evaluation, CNN/RNN and WaveNet methods stood out with their performance. CNN/RNN, trained on the normal subset of the DongTing dataset, appeared as the most prosperous method with an AUC value of 97.5%.

In [61], a dataset consisting of MFCC sequences was utilized to evaluate the performance of the SWMAT method. Using DTW to score the similarity between two MFCC sequences, SWMAT exhibited a performance of approximately 95% accuracy.

In [62], the performance of a hybrid model consisting of LSTM, NN, and RF methods was measured using the ADFA-LD dataset. In the preprocessing stage of the anomaly detection technique involving NN and RF methods, BoW, n-gram, and TFIDF were employed. The outputs of LSTM and the anomaly detection technique were processed by an ensemble method. The hybrid model reached an ACC of 97.2% and an FPR of 2.4% in the experiments.

E. WHAT ARE THE ADVANTAGES AND DISADVANTAGES OF ML/DL BASED METHODS USED IN HIDSs?

Most ML/DL based detection and preprocessing methods in Table 9 are commonly employed in HIDSs. Consequently, the performance of these detection and preprocessing methods directly influences the efficiency and effectiveness of the HIDS in which they are used. On the other hand, ML/DL based detection and preprocessing methods come with their advantages and disadvantages from different perspectives. For instance, while an ML/DL based detection method achieving high accuracy in identifying attacks is an advantage, the drawback lies in the time-consuming nature of the detection process. One way to reveal the merits and limitations of these methods is to examine their performance values in experiments. The ML/DL based methods in Table 9 are discussed according to the experimental performances outlined in Table 11. The analysis indicates that these methods generally detected attacks with high ACC rates. However, it is essential to note that certain methods possess distinct advantages or disadvantages. The specific advantages and disadvantages of these methods are listed as follows.

- The HMM method in [37] could be efficient regarding computational cost. However, this HMM method performed with low TPR values.
- In [38], the GBT method predicted the classes of attacks with the highest ACC rate. Nonetheless, the GBT method was the most time-consuming approach in predicting attack classes.
- The CNN method in [41] achieved high ACC rates on the CICIDS2017 and ADFA-LD datasets but exhibited low ACC rates on the NSL-KDD and ADFA-WD datasets. Additionally, the CNN method operated with high per-sample classification times.
- The PCA method in [42] was suitable for large-scale and high-dimensional spaces.
- The SVD dimension reduction and TFIDFvectorizer methods in [44] possess capabilities that might contribute to real-time attack detection. However, calculating TFIDF values created an additional computational overhead. Also, these methods achieved high FPR values in multi-classification experiments.
- The DL methods in [45] could face challenges in real-time applications due to long training and high detection times.
- The methods in [49] were not efficient in terms of time consumption. Consequently, converting these methods into real-time applications was a difficult task.
- The BoW TFIDF method in [52] contributed to detecting attacks with low FAR values.
- The K-means and KNN methods in [53] classified normal and U2L and PRB attack data samples in the NSL-KDD with low ACC rates.
- With or without using the DM feature selection method in [55], ML methods achieved low ACC rates on the NSL-KDD.
- The MLP and ReNN methods in [56] had the potential to be efficient in terms of resource consumption.
- The methods in [5], in experiments conducted on binary and multi-classification tasks, exhibited high FPR values.

TABLE 11. A summary of the experimental results.

Study	Detection Methods	Datasets	Number of Features Used	Experimental Results (Best Performances)
[36]	ML methods: OneR, DT, NB, BN, LR, RF, KNN, and SVM. Statistical methods: UGM and MGM.	The two datasets in [36].	13 or 15	SVM and RF achieved 100% ACC.
			15	UGM achieved 100% ACC
[37]	HMM.	Dataset of Purdue University.	-	0.1% FPR and 86.2% TPR.
[38]	DT, RF, GBT, SVM, MLP, and OCSVM.	The dataset in [38].	13	It created an additional overhead of 1.24% on the CPU and 1.87% on the memory. For anomaly detection, GBT achieved 100% ACC. For prediction time, DT had a time of 1.23 ms.
[40]	J48Consolidated.	NSL-KDD.	All.	99.962% ACC and 0.0004 FNR.
		ISCXIDS2012.	All.	99.936% ACC and 0.0006 FNR.
		CICIDS2017.	34	99.955% ACC and 0.0004 FNR.
[41]	CNN.	NSL-KDD.	30	83.43% ACC.
		CICIDS2017.	57	99.29% ACC.
		ADFA-LD.	-	95.34% ACC.
		ADFA-WD.	-	77.01% ACC.
[42]	LSTM, LSTM-PCA, and LSTM-MI.	KDD99.	2	LSTM-PCA with 99.44% ACC.
[44]	SVM, NN, and DT.	ADFA-LD.	-	In binary classification, SVM with 3.34% FPR. In multi-class classification, SVM with 9.12% FPR.
		ADFA-WD.	-	In binary classification, NN with 8.63% FPR. In multi-class classification, NN with 15.11% FPR.
[45]	WaveNet. LSTM. CNN/RNN. TLAD and ALAD.	ADFA-LD.	-	ALAD achieved a 99.8% AUC when combined with WaveNet.
		PLAID.	-	ALAD achieved a 99.5% AUC when combined with LSTM.
[49]	Non-temporal methods: NB, SVM, LR, and RF. Sequence-based methods: STM and LSTM+LR. Time-interval-based methods: TPA+NB, TPA+SVM, TPA+LR, TPA+RF, and TPF.	The dataset in [49].	-	For detecting unknown malware, TPA+LR achieved 99.6% ACC and 0.7% FPR. For detecting unknown types of malware, TPA+SVM achieved 89.28% ACC and 9.69% FPR. For classifying types of malware, TPA+LR achieved 99.24% ACC.
[52]	RF, J48, RIPPER, SVM, KNN, and NB.	ADFA-LD.	-	RF achieved 98.4% ACC and 1.7% FAR.
		VMM.	-	J48, RF, and RIPPER achieved 100% ACC and 0% FAR.

TABLE 11. *(Continued.)* **A summary of the experimental results.**

Study	Detection Methods	Datasets	Number of Features Used	Experimental Results (Best Performances)
[53]	K-means. KNN.	KDDCup99.	-	There was an improvement of at least 14.43% in terms of ACC and at least 12.3% in terms of REC.
		NSL-KDD.	-	There was an improvement of at least 9.94% in terms of ACC and at least 11.07% in terms of REC.
[55]	NN, DT, KNN, and Bagging.	NSL-KDD.	13	DT with 83.539% ACC.
		Combination of Kyoto and NSLKDD.	8	Bagging with 99.833% ACC.
[56]	MLP. ReNN.	BoT-IoT. TON-IoT. IoTID20.	-	Attacks were detected, with ACC rates reaching 100% on three datasets.
[5]	LSTM and Bi-LSTM. GRU and Bi-GRU. FCNN combining the outputs of LSTM and GRU methods.	ADFA-WD.	-	In the binary class, an ACC of 91.1% was achieved. In the multi-class, or, an ACC of 68.7% was achieved.
[57]	BPN.	KDDCup99.	-	96.5% DR.
[58]	ENBAG, ENKNN, ENBOS, and ENDSC.	ISCX-URL2016.	59	ENBAG achieved 99.3% and 97.9% ACC for binary and multi-classification, respectively.
[59]	CNN-DWA.	KDDCup99.	-	98.05% ACC.
[60]	CNN/RNN. LSTM. WaveNet. ECOD.	DongTing. ADFA-LD. PLAID.	-	CNN/RNN achieved 97.5% AUC.
[61]	SWMAT.	The dataset in [61].	-	Around 95% ACC.
[62]	LSTM. NN. RF.	ADFA-LD.	-	97.2% ACC.

- In [57], the BPN method using PSO achieved the best DR on the KDDCup99 dataset compared to other methods. However, BPN exhibited lower performance in the FAR metric than other methods.
- The ML methods in [58] were evaluated on the ISCX-URL2016 dataset using features selected by the MRMR algorithm. ENBAG showed high ACC values for binary and multi-class classification among these methods. In addition, ENKNN reached the highest detection time. In contrast, ENBOS and ENDSC classified URLs with low ACC values. Furthermore, ENDSC detected malicious URLs with a low ACC value.
- The CNN-DWA method in [59] was evaluated on KDDCup99 using label encoding in the preprocessing stage. CNN-DWA was more successful than CNN, with a higher ACC value.
- In [60], CNN/RNN and WaveNet methods achieved high AUC values on the DongTing dataset. However, ECOD exhibited lower performance with a low AUC on the DongTing dataset.
- The weighted average ensemble-based hybrid model consisting of LSTM, NN, and RF methods in [62] had high ACC and low FPR values. However, it showed less success regarding the FPR metric than a similar example in the literature.

VI. DISCUSSION AND FUTURE RESEARCH DIRECTIONS

The studies reviewed in this SLR are evaluated considering Tables 7 and 11. Based on the evaluation, the performances in the experiments of the detection methods used in the studies are discussed. In addition, defects in studies directly or indirectly related to HIDS and future research directions are identified. Below are evaluations of the performances of the detection methods.

- The ML methods in [36] demonstrated the capability to perform with high accuracy values; however, they carried the potential risk of high false alarm rates. In addition, statistical methods, trained with datasets that predominantly consist of normal data samples, tended to achieve lower false alarm rates than ML methods.
- In [37], the HMM represented a profile of normal user behaviors and identified deviations from this profile as anomalous. Therefore, HMM exhibited satisfactory performance in terms of the FPR. However, the method shouldn't have had low TPR values.
- Methods such as DT, RF, and GBT can achieve high accuracy rates on heterogeneous datasets. In this context, the GBT in [38], utilizing a heterogeneous dataset, detected attacks with high accuracy.
- In [40], J48Consolidated benefited from the effective performance of the CTC algorithm on datasets with class imbalance. Therefore, J48Consolidated achieved high accuracy values on imbalanced datasets such as NSL-KDD, ISCXIDS2012, and CICIDS2017.
- In [41], with its capability for multiple classifications, CNN performed well on the CICIDS2017 and ADFA-LD datasets by leveraging context-aware feature extraction in the preprocessing stage. However, it achieved lower accuracy rates on NSL-KDD and ADFA-WD datasets. The lack of developed CNN architectures suitable for these datasets could result in lower accuracy rates. Additionally, compared to others, the limited differences between normal and attack data examples in the ADFA-WD dataset impacted CNN's lower performance on ADFA-WD.
- In [42], using PCA in the preprocessing stage reduced the dataset's feature set dimension, resulting in the attainment of an optimal feature set. Therefore, LSTM with PCA achieved a high accuracy rate and decreased training time.
- In [43], the DSN difference was computed in the routing table of a network node, and it was discovered whether the node was suspicious by ACIDS. The decision about whether suspicious nodes were attack-related was based on the ID values in the RREP ID field. As a result, RREPs of attack-related nodes were discarded during the routing discovery process. Consequently, packet transmission rates increased, and the routing discovery process was secured. However, all these processes might introduce additional overhead in routing.
- In [44], n-gram feature vectors based on TFIDF values reduced by SVD were employed, enabling SVM on ADFA-LD and NN on ADFA-WD to detect anomalous system processes with low computational load and high accuracy values. However, these methods exhibited lower performance regarding the FPR metric.
- WaveNet contains discrete convolutions to capture context information. Therefore, in [45], ALAD with WaveNet performed best on ADFA-LD. On the other hand, WaveNet exhibited its worst performance on the PLAID dataset, possibly due to the likelihood of memorization during the training phase or the sensitive adjustment of architectural parameters.
- In [49], using temporal patterns of API calls during the training phase enabled detection methods to learn the malware behaviors better and improved their ability to differentiate from normal software. As a result, detection methods based on temporal patterns achieved high success values in detecting malware.
- In [52], the high performance of detection methods in terms of ACC and FAR metrics could be attributed to the use of various BoW methods in the preprocessing stage. Creating numerical vector representations containing frequency values of system calls by BoW methods was crucial in successfully learning patterns related to system calls by detection methods.
- In [53], DMAIDPS employed the distributed use of multiple agents to learn the behavior of the network. The distributed structure enhanced the learning rate and, consequently, improved the performance of detection methods.

- In [55], the DM method was suggested and utilized to select the most optimal features of datasets. Detection methods trained with features chosen by the DM method performed better at detecting attacks than methods in other studies.
- In [56], the underlying idea of SEHIDS was defined as the ability to train DL methods used as detectors repeatedly according to their performance values and to change the architectures of these methods constantly. Therefore, the performance of continuously updated detection methods in detecting attacks could remain at high rates.
- In [5], Word2Vec and Glove, word embedding methods, were used in the preprocessing stage to learn the contextual relationships of n-grams representing DLL calls. Providing word embedding vectors representing contextual relationships as input to DL-based detection methods increased the detection rate of these methods in identifying attacks.
- In [57], PSO performed feature extraction in the preprocessing stage. The BPN method, with these extracted features, detected system calls in KDDCup99 with a high DR. However, this method was less effective than other methods regarding the FAR metric.
- The ENBAG method in [58], compared to ENKNN, ENBOS, and ENDSC methods, demonstrated superior performance in both binary and multi-class classification regarding the ACC metric. Additionally, ENKNN exhibited the lowest detection time, showcasing its suitability for high-bandwidth networks.
- The proposed CNN-DWA method in [59] was designed to overcome feature selection and imbalanced data challenges. As a result, CNN-DWA achieved a high ACC value on the imbalanced KDDCup99 dataset.
- Among the methods used to evaluate the DongTing dataset in [60], CNN/RNN emerged as the most successful, while ECOD performed the least successfully.
- The proposed SWMAT technique in [61] transformed dynamic memory traces into sound wave signals, extracted MFCCs, and determined similarities between MFCC sequences. SWMAT computed similarities between MFCC sequences with a high accuracy value. However, the DTW computation used in SWMAT's similarity computation process took a long time.
- The hybrid model containing LSTM, NN, and RF methods in [62] showed high performance in terms of ACC metric on the ADFA-LD dataset since it contains an ensemble-based method. On the other hand, this hybrid model had a worse FPR value than the model in a similar study in the literature.

The defects in the studies and the future research directions are given below.

- Except for [38] and [61], the proposed HIDSs or IDSs in the studies were not tested in a real environment. In other words, the efficiency of the proposed HIDSs or IDSs in the literature was not evaluated under real attack scenarios and natural settings. Consequently, evaluating the performance of the HIDSs or IDSs in real-world environments and time frames remains an area of future research.
- It is evident that the proposed HIDSs or IDSs in the studies, except for [53], did not include attack prevention functions. Therefore, adding attack prevention functionalities to HIDSs or IDSs can be a potential focus for future research.
- New datasets were created in [36], [38], [45], [49], [60], and [61] studies. However, according to Table 10, most HIDS or IDS related studies utilized popular datasets. Nonetheless, studies that created their datasets had not publicly shared them. The lack of datasets containing up-to-date normal and attack data samples relevant to HIDSs or IDSs is identified as a problem.
- The CPU and memory overhead of the HIDS or IDSs proposed in studies other than [38] and [56] were not evaluated. Therefore, measuring the efficiency of future introduced HIDSs or IDSs in terms of resource consumption offers an alternative evaluation option.
- In studies other than [40], [41], [45], [56], [58], and [60], detection, testing, classification or inference times were not measured. Therefore, the time it takes for HIDS or IDS to detect attacks and identify the types of attacks stand as criteria that will be useful in evaluating the suitability of these systems for real-time applications.
- In studies other than [52], both 100% ACC rates and 0% FAR values could not be reached simultaneously. In the real environment, detecting attacks at these rates or values is nearly impossible. However, detecting attacks with high ACC rates and low FAR values is among the objectives of future studies.

On the other hand, the importance and needs of the noticed HIDSs during the review of studies are provided below in bullet points.

Importance:

- **Intrusion Detection:** HIDSs monitor various activities, such as system calls and logs on main computers, collecting relevant data. Subsequently, they analyze the collected data using various methods, such as ML/DL, to detect whether there is any attack or threat.
- **Protection Against Internal or External Attacks:** HIDSs advance the security level of hosts against potential attacks from internal and external sources. Malicious users or authorized individuals with access to hosts can be responsible for internal attacks, while an unknown source can initiate external attacks.
- **Early Warning System and Damage Mitigation:** HIDSs issue alarms when attacks on hosts begin, providing early warnings to system administrators. These early warnings prevent and limit attacks on the hosts from causing significant damage.
- **Recording and Reviewing of Activities:** HIDSs record various information related to the activities of hosts.

Reviewing the recorded information can contribute to the discovery of potential security vulnerabilities on hosts.

Needs:

- **Data Source:** HIDSs require various activity data from hosts, such as system calls, DLL files, system logs, and registry keys, to detect attacks. These data must be accurate and accessible by HIDSs.
- **Configuration and Customization:** Identifying activities on main computers as attacks or non-attacks depends on the usage environment of the hosts. Therefore, HIDSs need to be developed and customized to adapt to the usage environments of hosts. Detection methods in HIDSs should be configured to be specific to the environments of hosts, capable of determining whether activities are attack-related.
- **Real-time Monitoring and Alerting:** To prevent extensive damage to the hosts, HIDSs need to monitor and analyze activities in real-time and alert system administrators during any attack.
- **Analysis and Correlation:** HIDSs must correlate activities related to hosts with each other and analyze them accurately. In analysis, FPR should be low to detect actual attacks and avoid providing system administrators with incorrect alerts.
- **Reporting and Logging:** HIDSs should generate detailed reports and log data on activities related to hosts. Examining reports and log data of activities is crucial in detecting security vulnerabilities on hosts.
- **Update:** Due to the emergence of different attacks or changes in existing attack types, HIDSs must regularly update their databases and detection methods.

VII. CONCLUSION

In this SLR, 21 studies published between 2020 and 2023, which were generally centered around HIDS, were examined. Five research questions were addressed in the review of the studies. In the context of these research questions, the metrics preferred to evaluate the performance of HIDSs were discussed. Subsequently, the detection methods in HIDSs were emphasized, and the performance values of these methods were investigated. Following this, the focus was on the datasets used in the training and testing ML/DL methods employed as detection methods. Additionally, the advantages and disadvantages of ML/DL based detection methods were identified. After the research questions were answered, the performances of the detection methods used in the studies were evaluated. On the other hand, deficiencies in the examined studies were identified. In light of these deficiencies, potential future research directions were discussed. The identified deficiencies were listed as follows:

- The proposed HIDS or IDS were not generally tested in real-world environments.
- The functionality of attack prevention was not included in many HIDS or IDS.
- Outdated datasets were commonly used when developing learning-based HIDS or IDS.
- Assessments regarding additional overhead on CPU and memory were usually overlooked.
- Many HIDS or IDS were not evaluated regarding detection and classification times.
- Generally, the balance of detecting attacks with both high ACC and low FAR values was not achieved by HIDSs or IDSs.

Finally, the importance and needs of HIDSs were defined from various perspectives. In future research, it is aimed to develop a HIDS constructed with up-to-date data samples, tested in real-world environments, and including the functionality of attack prevention.

REVIEWER APPRECIATION

The authors would like to express their sincere gratitude to the reviewers for their meticulous evaluation and invaluable feedback, which significantly contributed to the enhancement of the quality and depth of this study.

REFERENCES

[1] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 16–24, 2013.
[2] I. Martins, J. S. Resende, P. R. Sousa, S. Silva, L. Antunes, and J. Gama, "Host-based IDS: A review and open issues of an anomaly detection system in IoT," *Future Gener. Comput. Syst.*, vol. 133, pp. 95–113, Aug. 2022.
[3] A. Thakkar and R. Lohiya, "A survey on intrusion detection system: Feature selection, model, performance measures, application perspective, challenges, and future research directions," *Artif. Intell. Rev.*, vol. 55, no. 1, pp. 453–563, Jan. 2022.
[4] P. K. Keserwani, M. C. Govil, and E. S. Pilli, "An effective NIDS framework based on a comprehensive survey of feature optimization and classification techniques," *Neural Comput. Appl.*, vol. 35, no. 7, pp. 4993–5013, Mar. 2023.
[5] Y. Kumar and B. Subba, "Stacking ensemble-based HIDS framework for detecting anomalous system processes in windows based operating systems using multiple word embedding," *Comput. Secur.*, vol. 125, Feb. 2023, Art. no. 102961.
[6] I. Corona, G. Giacinto, and F. Roli, "Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues," *Inf. Sci.*, vol. 239, pp. 201–225, Aug. 2013.
[7] K. He, D. D. Kim, and M. R. Asghar, "Adversarial machine learning for network intrusion detection systems: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 1, pp. 538–566, 1st Quart., 2023.
[8] M. R. Ayyagari, N. Kesswani, M. Kumar, and K. Kumar, "Intrusion detection techniques in network environment: A systematic review," *Wireless Netw.*, vol. 27, no. 2, pp. 1269–1285, Feb. 2021.
[9] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, "Survey on SDN based network intrusion detection system using machine learning approaches," *Peer-to-Peer Netw. Appl.*, vol. 12, no. 2, pp. 493–501, 2019.
[10] V. Jyothsna, R. Prasad, and K. M. Prasad, "A review of anomaly based intrusion detection systems," *Int. J. Comput. Appl.*, vol. 28, no. 7, pp. 26–35, 2011.
[11] V. Bukac, P. Tucek, and M. Deutsch, "Advances and challenges in standalone host-based intrusion detection systems," in *Proc. 9th Int. Conf. Trust, Privacy Secur. Digit. Bus.*, Vienna, Austria. Berlin, Germany: Springer, 2012, pp. 105–117.
[12] S. Jose, D. Malathi, B. Reddy, and D. Jayaseeli, "A survey on anomaly based host intrusion detection system," *J. Phys., Conf. Ser.*, vol. 1000, Apr. 2018, Art. no. 012049.
[13] Y. Shin and K. Kim, "Comparison of anomaly detection accuracy of host-based intrusion detection systems based on different machine learning algorithms," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 2, pp. 252–259, 2020.

[14] A. S. Ashoor and S. Gore, "Importance of intrusion detection system (IDS)," *Int. J. Sci. Eng. Res.*, vol. 2, no. 1, pp. 1–4, 2011.

[15] J. L. Leevy and T. M. Khoshgoftaar, "A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 big data," *J. Big Data*, vol. 7, no. 1, pp. 1–19, Dec. 2020.

[16] G. Kocher and G. Kumar, "Machine learning and deep learning methods for intrusion detection systems: Recent developments and challenges," *Soft Comput.*, vol. 25, no. 15, pp. 9731–9763, Aug. 2021.

[17] D. Schubert, H. Eikerling, and J. Holtmann, "Application-aware intrusion detection: A systematic literature review, implications for automotive systems, and applicability of AutoML," *Frontiers Comput. Sci.*, vol. 3, Aug. 2021, Art. no. 567873.

[18] H. A. Hassan, E. E. Hemdan, W. El-Shafai, M. Shokair, and F. E. A. El-Samie, "Intrusion detection systems for the Internet of Thing: A survey study," *Wireless Pers. Commun.*, vol. 128, no. 4, pp. 2753–2778, Feb. 2023.

[19] A. Heidari and M. A. Jabraeil Jamali, "Internet of Things intrusion detection systems: A comprehensive review and future directions," *Cluster Comput.*, vol. 26, no. 6, pp. 3753–3780, Dec. 2023.

[20] M. Zipperle, F. Gottwalt, E. Chang, and T. Dillon, "Provenance-based intrusion detection systems: A survey," *ACM Comput. Surv.*, vol. 55, no. 7, pp. 1–36, Dec. 2022.

[21] A. Thakkar and R. Lohiya, "A review on challenges and future research directions for machine learning-based intrusion detection system," *Arch. Comput. Methods Eng.*, vol. 30, no. 7, pp. 4245–4269, Sep. 2023.

[22] Z. T. Sworna, Z. Mousavi, and M. A. Babar, "NLP methods in host-based intrusion detection systems: A systematic review and future directions," *J. Netw. Comput. Appl.*, vol. 220, Nov. 2023, Art. no. 103761.

[23] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41525–41550, 2019.

[24] A. Almomani, M. Alauthman, F. Albalas, O. Dorgham, and A. Obeidat, "An online intrusion detection system to cloud computing based on NeuCube algorithms," in *Cognitive Analytics: Concepts, Methodologies, Tools, and Applications*. Hershey, PA, USA: IGI Global, 2020, pp. 1042–1059.

[25] J. M. Vidal, M. A. S. Monge, and S. M. M. Monterrubio, "Anomaly-based intrusion detection: Adapting to present and forthcoming communication environments," in *Handbook of Research on Machine and Deep Learning Applications for Cyber Security*. Hershey, PA, USA: IGI Global, 2020, pp. 195–218.

[26] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1–22, Dec. 2019.

[27] A. Abusitta, M. Q. Li, and B. C. M. Fung, "Malware classification and composition analysis: A survey of recent developments," *J. Inf. Secur. Appl.*, vol. 59, Jun. 2021, Art. no. 102828.

[28] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in Internet of Things," *J. Netw. Comput. Appl.*, vol. 84, pp. 25–37, Apr. 2017.

[29] J. R. Vacca, *Computer and Information Security Handbook*. Waltham, MA, USA: Newnes, 2012.

[30] F. Sabahi and A. Movaghar, "Intrusion detection: A survey," in *Proc. 3rd Int. Conf. Syst. Netw. Commun.*, Oct. 2008, pp. 23–26.

[31] S. Agrawal and J. Agrawal, "Survey on anomaly detection using data mining techniques," *Proc. Comput. Sci.*, vol. 60, pp. 708–713, Jan. 2015.

[32] H. Hindy, D. Brosset, E. Bayne, A. K. Seeam, C. Tachtatzis, R. Atkinson, and X. Bellekens, "A taxonomy of network threats and the effect of current datasets on intrusion detection systems," *IEEE Access*, vol. 8, pp. 104650–104675, 2020.

[33] V. Hajisalem and S. Babaie, "A hybrid intrusion detection system based on ABC-AFS algorithm for misuse and anomaly detection," *Comput. Netw.*, vol. 136, pp. 37–50, May 2018.

[34] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2015.

[35] L. N. Tidjon, M. Frappier, and A. Mammar, "Intrusion detection systems: A cross-domain overview," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3639–3681, 4th Quart., 2019.

[36] J. Ribeiro, F. B. Saghezchi, G. Mantas, J. Rodriguez, S. J. Shepherd, and R. A. Abd-Alhameed, "An autonomous host-based intrusion detection system for Android mobile devices," *Mobile Netw. Appl.*, vol. 25, no. 1, pp. 164–172, 2020.

[37] J. Liu, M. Duan, W. Li, and X. Tian, "HMMs based masquerade detection for network security on with parallel computing," *Comput. Commun.*, vol. 156, pp. 168–173, Apr. 2020.

[38] R. Gassais, N. Ezzati-Jivan, J. M. Fernandez, D. Aloise, and M. R. Dagenais, "Multi-level host-based intrusion detection system for Internet of Things," *J. Cloud Comput.*, vol. 9, no. 1, pp. 1–16, Dec. 2020.

[39] M. Desnoyers and M. R. Dagenais, "The LTTng tracer: A low impact performance and behavior monitor for GNU/Linux," in *Proc. Ottawa Linux Symp.*, 2006, pp. 209–224.

[40] R. Panigrahi, S. Borah, A. K. Bhoi, M. F. Ijaz, M. Pramanik, Y. Kumar, and R. H. Jhaveri, "A consolidated decision tree-based intrusion detection system for binary and multiclass imbalanced datasets," *Mathematics*, vol. 9, no. 7, p. 751, Mar. 2021.

[41] E. A. Shams, A. Rizaner, and A. H. Ulusoy, "A novel context-aware feature extraction method for convolutional neural network-based intrusion detection systems," *Neural Comput. Appl.*, vol. 33, no. 20, pp. 13647–13665, Oct. 2021.

[42] F. Laghrissi, S. Douzi, K. Douzi, and B. Hssina, "Intrusion detection systems using long short-term memory (LSTM)," *J. Big Data*, vol. 8, no. 1, p. 65, Dec. 2021.

[43] S. Sivanesh and V. R. S. Dhulipala, "Accurate and cognitive intrusion detection system (ACIDS): A novel black hole detection mechanism in mobile ad hoc networks," *Mobile Netw. Appl.*, vol. 26, no. 4, pp. 1696–1704, Aug. 2021.

[44] B. Subba and P. Gupta, "A tfidfvectorizer and singular value decomposition based host intrusion detection system framework for detecting anomalous system processes," *Comput. Secur.*, vol. 100, Jan. 2021, Art. no. 102084.

[45] J. H. Ring, C. M. Van Oort, S. Durst, V. White, J. P. Near, and C. Skalka, "Methods for host-based intrusion detection with deep learning," *Digit. Threats, Res. Pract.*, vol. 2, no. 4, pp. 1–29, Dec. 2021.

[46] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," 2016, *arXiv:1609.03499*.

[47] G. Kim, H. Yi, J. Lee, Y. Paek, and S. Yoon, "LSTM-based system-call language modeling and robust ensemble method for designing host-based intrusion detection systems," 2016, *arXiv:1611.01726*.

[48] A. Chawla, B. Lee, S. Fallon, and P. Jacob, "Host based intrusion detection system with combined CNN/RNN model," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, Dublin, Ireland. Cham, Switzerland: Springer, 2018, pp. 149–158.

[49] I. Finder, E. Sheetrit, and N. Nissim, "Time-interval temporal patterns can beat and explain the malware," *Knowl.-Based Syst.*, vol. 241, Apr. 2022, Art. no. 108266.

[50] *Cuckoo Sandbox*. Accessed: Feb. 15, 2024. [Online]. Available: https://cuckoosandbox.org/

[51] *VirusTotal*. Accessed: Feb. 15, 2024. [Online]. Available: https://www.virustotal.com/gui/home/upload

[52] A. A. R. Melvin, G. J. W. Kathrine, S. Pasupathi, V. Shanmuganathan, and R. Naganathan, "An AI powered system call analysis with bag of word approaches for the detection of intrusions and malware in Australian defence force academy and virtual machine monitor malware attack data set," *Expert Syst.*, May 2022, Art. no. e13029.

[53] A. Javadpour, P. Pinto, F. Ja'fari, and W. Zhang, "DMAIDPS: A distributed multi-agent intrusion detection and prevention system for cloud IoT environments," *Cluster Comput.*, vol. 26, no. 1, pp. 367–384, Feb. 2023.

[54] Y. Xiao and X. Xiao, "An intrusion detection system based on a simplified residual network," *Information*, vol. 10, no. 11, p. 356, Nov. 2019.

[55] P. Ghosh, S. Sinha, R. R. Sharma, and S. Phadikar, "An efficient IDS in cloud environment using feature selection based on DM algorithm," *J. Comput. Virol. Hacking Techn.*, vol. 18, no. 3, pp. 243–258, Sep. 2022.

[56] M. Baz, "SEHIDS: Self evolving host-based intrusion detection system for IoT networks," *Sensors*, vol. 22, no. 17, p. 6505, Aug. 2022.

[57] G. Nagarajan and P. J. Sajith, "Optimization of BPN parameters using PSO for intrusion detection in cloud environment," *Soft Comput.*, pp. 1–12, Jun. 2023.

[58] Q. A. Al-Haija and M. Al-Fayoumi, "An intelligent identification and classification system for malicious uniform resource locators (URLs)," *Neural Comput. Appl.*, vol. 35, no. 23, pp. 16995–17011, Aug. 2023.

[59] A. K. Samha, N. Malik, D. Sharma, and P. Dutta, "Intrusion detection system using hybrid convolutional neural network," *Mobile Netw. Appl.*, pp. 1–13, Aug. 2023.

[60] G. Duan, Y. Fu, M. Cai, H. Chen, and J. Sun, "DongTing: A large-scale dataset for anomaly detection of the Linux kernel," *J. Syst. Softw.*, vol. 203, Sep. 2023, Art. no. 111745.
[61] R. Vijayakanthan, I. Ahmed, and A. Ali-Gombe, "SWMAT: Mel-frequency cepstral coefficients-based memory fingerprinting for IoT devices," *Comput. Secur.*, vol. 132, Sep. 2023, Art. no. 103298.
[62] A. Chaudhari, B. Gohil, and U. P. Rao, "A novel hybrid framework for cloud intrusion detection system using system call sequence analysis," *Cluster Comput.*, pp. 1–17, Nov. 2023.
[63] Z. Li, Y. Zhao, X. Hu, N. Botta, C. Ionescu, and G. Chen, "ECOD: Unsupervised outlier detection using empirical cumulative distribution functions," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12181–12193, Dec. 2022.
[64] T. Lane and C. E. Brodley, "An application of machine learning to anomaly detection," in *Proc. 20th Nat. Inf. Syst. Secur. Conf.*, vol. 377, Baltimore, MD, USA, 1997, pp. 366–380.
[65] W. Elmasry, A. Akbulut, and A. H. Zaim, "Deep learning approaches for predictive masquerade detection," *Secur. Commun. Netw.*, vol. 2018, pp. 1–24, Aug. 2018.
[66] *KDD Cup 1999 Data*. Accessed: Feb. 15, 2024. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[67] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, Jul. 2009, pp. 1–6.
[68] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Secur.*, vol. 31, no. 3, pp. 357–374, 2012.
[69] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. ICISSP*, vol. 1, 2018, pp. 108–116.
[70] G. Creech and J. Hu, "Generation of a new IDS test dataset: Time to retire the KDD collection," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2013, pp. 4487–4492.
[71] G. Creech, "Developing a high-accuracy cross platform host-based intrusion detection system capable of reliably detecting zero-day attacks," Ph.D. dissertation, UNSW Sydney, Sydney, NSW, Australia, 2014.
[72] A. A. R. Melvin, G. J. W. Kathrine, S. S. Ilango, S. Vimal, S. Rho, N. N. Xiong, and Y. Nam, "Dynamic malware attack dataset leveraging virtual machine monitor audit data for the detection of intrusions in cloud," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 4, p. e4287, Apr. 2022.
[73] 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," in *Proc. 1st Workshop Building Anal. Datasets Gathering Exper. Returns Secur.*, Apr. 2011, pp. 29–36.
[74] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Gener. Comput. Syst.*, vol. 100, pp. 779–796, Nov. 2019.
[75] I. Ullah and Q. H. Mahmoud, "A scheme for generating a dataset for anomalous activity detection in IoT networks," in *Proc. Can. Conf. Artif. Intell.* Cham, Switzerland: Springer, 2020, pp. 508–520.
[76] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustain. Cities Soc.*, vol. 72, Sep. 2021, Art. no. 102994.
[77] *ISCX-URL2016*. Accessed: Feb. 15, 2024. [Online]. Available: https://www.unb.ca/cic/datasets/url-2016.html

HAMİ SATILMIŞ received the B.Sc. degree in computer engineering from Eskişehir Osmangazi University, Eskişehir, Turkey, in 2016, and the M.Sc. degree in computer engineering from Ondokuz Mayıs University, Samsun, Turkey, in 2020, where he is currently pursuing the Ph.D. degree with the Department of Computational Science. He is also a Research Assistant with the Department of Computer Engineering, Ondokuz Mayıs University. His research interests include post-quantum cryptography, information security, machine learning, deep learning, and software engineering.

SEDAT AKLEYLEK received the B.Sc. degree in mathematics majored in computer science from Ege University, izmir, Turkey, in 2004, and the M.Sc. and Ph.D. degrees in cryptography from Middle East Technical University, Ankara, Turkey, in 2008 and 2010, respectively. He was a Postdoctoral Researcher with the Cryptography and Computer Algebra Group, TU Darmstadt, Germany, from 2014 to 2015. He was a Professor with the Department of Computer Engineering, Ondokuz Mayıs University, Samsun, Turkey. He has been a Professor with the Department of Computer Engineering, Istinye University, Istanbul, Turkey. He has been with the Chair of Security and Theoretical Computer Science, University of Tartu, Tartu, Estonia, since 2022. His research interests include the areas of post-quantum cryptography, algorithms and complexity, architectures for computations in finite fields, applied cryptography for cyber security, malware analysis, the IoT security, and avionics cyber security. He is an Editorial Board Member of IEEE Access, *Turkish Journal of Electrical Engineering and Computer Sciences*, *PeerJ Computer Science*, and *International Journal of Information Security Science*.

ZALİHA YÜCE TOK received the B.S. degree in computer science and the M.S. and Ph.D. degrees in cryptography from Middle East Technical University, Ankara, Turkey, in 2004, 2007, and 2016, respectively. She is currently an Avionic Software Engineer with the ASELSAN Avionic Cyber Security Frontier Laboratory. Her current research interests include post quantum cryptography, applied cryptography for cyber security, and avionic cyber security.