A Holistic Review of Network Anomaly Detection Systems: A Comprehensive Survey

Nour Moustafa[a,*], Jiankun Hu[a], Jill Slay[b]

[a] *School of Engineering and Information Technology, University of New South Wales at ADFA, Northcott Dr, Campbell ACT 2612, Canberra, Australia*
[b] *La Trobe University, Melbourne, Australia*

Abstract

Network Anomaly Detection Systems (NADSs) are gaining a more important role in most network defense systems for detecting and preventing potential threats. The paper discusses various aspects of anomaly-based Network Intrusion Detection Systems (NIDSs). The paper explains cyber kill chain models and cyber-attacks that compromise network systems. Moreover, the paper describes various Decision Engine (DE) approaches, including new ensemble learning and deep learning approaches. The paper also provides more details about benchmark datasets for training and validating DE approaches. Most of NADSs' applications, such as Data Centers, Internet of Things (IoT), as well as Fog and Cloud Computing, are also discussed. Finally, we present several experimental explanations which we follow by revealing various promising research directions.

Keywords: Intrusion Detection system (IDS), Network Anomaly Detection Systems (NADS), data pre-processing, Decision Engine (DE)

1. Introduction

An Intrusion Detection System (IDS) is important in the cyber security field for achieving a solid line of protection against cyber adversaries. The digital world has become the main complement of the physical world because of the prevalent use of computer and network systems and their IoT services that easily execute users' tasks in a short time and at low cost. Since means of information technology are rapidly spreading throughout the world, the need for securing network resources against cyber threats has been increasing. Some of the existing technologies are not securely designed, so it is essential to consider *security by design* for protecting them.

A system is treated secure if the three principles of computer security, Confidentiality, Integrity and Availability (CIA), are successfully achieved [1, 2, 3, 4, 5]. Every attacker has its own complex techniques, which poses serious threats to computer networks. When an attacker gathers significant information about

*Corresponding author
Email addresses: nour.moustafa@unsw.edu.au (Nour Moustafa), J.Hu@adfa.edu.au (Jiankun Hu), J.Slay@latrobe.edu.au (Jill Slay)

a system, it breaches the system's confidentiality and, when it interrupts legitimate operations, it compromises its availability and integrity. For example, Denial of Service (DoS) attack disrupts client systems, which breaches the availability principle, while malware code hijacks the program's implementation which violates the integrity principle [3, 6, 7].

An IDS is a technique for monitoring and inspecting the activities that take place in a computer or network system to detect possible threats by measuring their violations of computer security principles of CIA [8, 9, 10, 11]. The classical architecture of a Network IDS (NIDS) comprises four components [12], as shown in Figure 1, namely, a packet decoder, pre-processor, DE sensor and defence response/alert module, as described below.

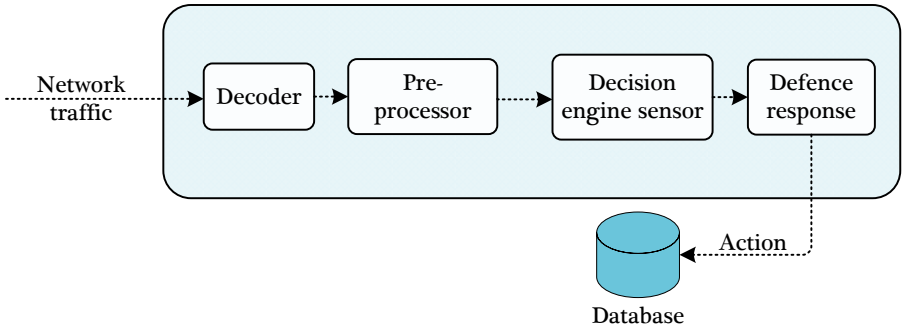

Figure 1: Architecture of classical IDS

- The packet decoder acquires portions of raw network traffic using audit data collection tools, such as Tcpdump and Libpcap, which transfer each portion into the pre-processor for handling.
- The pre-processor captures a set of features from the raw audit data which is used later in the DE sensor. A typical pre-processor is the TCP handler which analyses TCP protocols in session flows; for example, Netflow, Bro-IDS and Argus tools which examine different protocols, such as HTTP, DNS, SMTP and UDP.
- The DE sensor receives the extracted features from the pre-processor and builds a model that distinguishes attack observations from normal ones. If an attack is detected, it requests the defence response for raising an alert.
- The defence response refers to the following activities: (i) a DE triggers alerts and logs them in a database, and (ii) the DE sends the alerts to a security administrator for making an action.

Over the last decade, there are many surveys that have been conducted for reviewing the IDS technology. Chandola et al. [13] discussed the foundations of anomaly detection approaches and their applicability in different domains. Garcia-Teodoro et al. [11] reviewed anomaly detection methods of statistical, knowledge, machine learning, as well as their issues. Ahmed et al. [14] described

the methods of anomaly detection systems and some challenges of IDS datasets. In [15], hybrid IDSs were discussed by integrating feature selection and detection methods for improving the detection accuracy, but they have a drawback of demanding highly computational resources. Peng et al. [16] discussed intrusion detection and prevention techniques by designing user profiles and discovering variations as anomalies. Recently, researchers surveyed the deployment of IDSs in different applications such as Internet of Things (IoT)-based IDS [17] and Cloud-based IDS [18]. For example, Zarpelao et al. [10] presented a review of IDSs in IoT networks. The authors described detection approaches, IDS deployments, and security threats. Sharma et al. [19] explained the methodologies of deploying IDSs in VANET and VANET Cloud. Recently, Resende and Drummond [20] presented a comprehensive discussion of using Random Forest methods for developing a reliable IDS. Although the existing surveys discussed various aspects of IDSs, our survey provides a holistic review that gives a better understanding of designing anomaly detection in different domains.

The main contributions of this survey include the following.

- We provide a comprehensive discussion of network threats and intrusion detection properties.
- We describe an architecture for the Network Anomaly Detection System (NADS) with describing its components.
- We explain the recent methodologies, involving ensemble- learning and deep-learning algorithms, and challenges of designing an effective NADS.
- We conduct several experiments using different network datasets, feature selection and DE techniques to demonstrate their applicability for evaluating NADSs.

The remainder of this paper is organised as follows. Section 2 explains contemporary network threats and attacks detected by IDSs. The properties of IDSs are discussed in Section 3 while the components of NADS are presented in Section 4. DE approaches are discussed in Section 5. Section 6 outlines the evaluation metrics used for IDSs. Practical insights of feature selection and DE evaluations are provided in Section 7. Section 8 describes the challenges and future directions of NADSs. Finally, concluding remarks are introduced in Section 9.

2. Contemporary network threats

The numbers, types and complexities of network threats are increasing. Cyber adversaries can cause financial losses and reputational damage, steal sensitive information and intellectual property, and interrupt business. Since attacks have become more complex, including a set of stealthy and sophisticated hacking processes called an Advanced Persistent Threat (APT), the APT Intrusion Kill Chain security model has become popular to describe the stages of attacks [21]. The APT Intrusion Kill Chain relies on the premise that an attack has an operational life cycle for gathering information and exploiting the victim system. The steps in the chain relate to recent anomalous events covering a set of common actions in a targeted attack. A better understanding of the cyber kill

Table 1: Attacks against computer and network systems could be identified by NIDSs

Attack types	**Properties**	**Examples**
Information Gathering and Probing	- scan computer and network systems to find vulnerabilities - provide lists of vulnerabilities, such as SMBv1 and open ports, to an attacker for exploiting victims	IPsweep, portsweep, SYS scan, FIN scan
User to Root (U2R)	- can breach vulnerabilities to gain privileges of a system's superuser while starting as a legitimate user	Rootkit, loadmodule
Remote to Local (R2L)	- can transmit packets to a remote system over a network without having an account on that system and gain access to harm the system's operations.	Warezclient, warezmaste, spy
Malware	- includes any executable malicious scripts like worms and viruses	SQL Slammer worms, Tuareg viruses
Flooding attacks	- contain malicious events that massively transmit superfluous requests for disrupting computer resources such as DoS and Distrusted DoS (DDoS)	Buffer overflow, TCP SYN, teardrop, smurf

chain's life cycle assists in designing an effective and reliable NADS that can efficiently discover existing and future malicious activities [22].

An attacker's philosophy almost invariably comprises two phases [22]. The first, the so-called exploitation phase, is a method for controlling the execution flow in the targeted program. At its abstract level, this can be a stack/heap-based buffer overflow in which an intrusively long text overwrites the instruction pointers of the targeted program but also includes a full suite of methods which can be used by more sophisticated adversaries to gain control of a system while its code is running. The second phase is known as the payload phase. After successfully exploiting the execution flow to the payload, this phase performs the aim of the attacker, such as to steal information and/or disrupt computer resources. The payload process is executed through a shellcode terminal which establishes a command prompt on the hacker's computer to execute post-exploitation events. Existing IDSs can identify attack types listed in Table 1 if their DE approaches are well-designed [23, 24]. Based on the Australian Cyber Security Centre (ACSC) [25], McAfee threat reports [26], Figure 2 depicts the current variants of attacks which still expose computer networks and require further research to be discovered using NADSs, as detailed in the following.

- **A DoS** is an attempt by an attacker to prevent legitimate access to websites by overwhelming the amount of available bandwidth or resources of the computer system (e.g., zombies). When many computer systems are utilised to investigate such activities, such as applying a botnet, it is known as a DDoS attack. The number of these network attacks has been increasing, with a variety of DDoS types of attack sending more than 100 Gbps which constitute serious vulnerabilities for computer networking [27].

- **Brute Force** endeavours to illegally obtain pairs of user names and passwords by trying all predefined pairs to gain access to network services, with automated applications often used to guess password combinations.

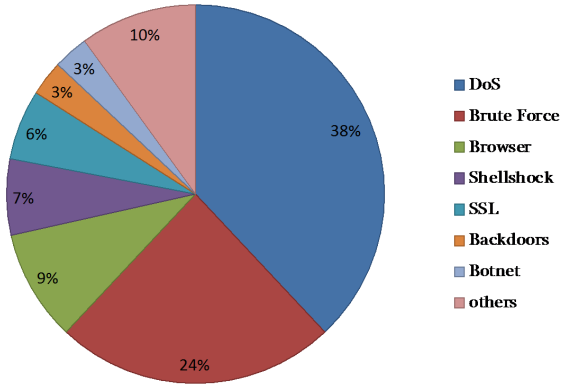

Figure 2: Recent top network attacks

To prevent such an attack, network administrators can place restrictions on the acceptable number of login attempts and generate a blacklist for a client whose network traffic are anomalous. This leads to the blocking of IP addresses after multiple login failures as well as limiting access to specific IP addresses [28].

- **Browser-based network attacks**, such as Tor, attempt to penetrate anonymous communication systems by exploiting JavaScript and HTML, such as Cross-site Scripting (XSS), to create some predefined rules for correlating user activities based on the websites visited. They are often executed by an attacker penetrating a client's vulnerabilities, which are typically triggered by outdated software, and possibly tempting the user to unwittingly download malware masquerading as a fake software/application update. A common solution to browser-based attacks is to frequently update web browsers and their services, for example, Java and Flash, so that browser vulnerabilities are easily detected [26, 29].

- **Shellshock attacks** relate to vulnerabilities that breach the command-line shell of Linux, UNIX and Apple OS systems called Bash. When Shellshock appeared in September 2014, many computer systems and appliances were vulnerable as they could be penetrated by a remote code execution which possibly authorised attackers to have full access and control. This permitted anomalous commands to be executed which could then download and implement anomalous scripts.

- **A successful SSL attacker** aims to intercept encrypted data, send them over a network and then access the unencrypted data and benefit by gaining access to applications. In April 2014, a dangerous vulnerability in the OpenSSL execution of the TLS/SSL Heartbeat extension, namely Heartbleed, was publicly released and caused the leaking of memory data. An

attacker could also access private keys, confidential information and secure content which could help other cyber adversaries. Moreover, these vulnerabilities allowed attackers to continually access the private information in systems by sending a wide variety of malicious commands to susceptible servers [25, 26].

- **A backdoor attack** can be defined as a technique which exposes computers to remote access by naturally replying to particularly constructed client applications. Several of them essentially use the IRC backbone and receive commands from IRC chat clients through the IRC network [26, 30]. They are less popular attacks than others and often used as part of targeted intrusions [30] which can be custom-designed to evade security detection and provide a masked point of entry.

- **A botnet** denotes the number of hijacked computer systems remotely operated by one or many malicious actors which coordinate their activities by Command and Control (C&C). Networks are regularly hit with attempts to expose their computer systems and appliances as attackers execute DDoS attacks to send spam email or implement fraudulent botnets to penetrate their targeted networks [25, 26].

3. Intrusion detection properties

An IDS can be categorised into five ways: monitored environments; detection approaches; applications and deployments; anomaly types; and defence responses [2, 12, 31, 32], as discussed in the following.

3.1. Monitored environments

An IDS can be used to monitor host- and network-based environments. Firstly, a host-based IDS (HIDS) monitors the events of a host by collecting information about activities which happen in a computer system. A sensor should be installed in such a system to monitor hosts and log the operating system's activities [33]. Secondly, a Network-based IDS (NIDS) monitors network traffic to identify remote attacks that happen over a network connection [4, 12, 34]. A NIDS has always been an essential security solution as it provides a solid line of defence against a malicious activity before it accesses the resources of a host and records itself in the audit trials of an operating system. Although a HIDS can detect intrusions into hosts, this naturally occurs after a host's computer resources, such as its files and services, are accessed. It is clear that the best security solution is to deter known and zero-day attacks before they exploit hosts, i.e., over networks, to achieve the wisdom of 'prevention is better than cure'.

A modern NIDS can deal with end-to-end encryption by extracting general and statistical information about packets, for instance, their sizes, lengths and inter-arrival times, as flow-based features [33, 35] but packet payloads, namely, packet-based features, always obfuscate. These packets have been analysed using Deep Packet Inspection (DPI) paradigms, with their classifications based on their behaviours or a hybrid of learning theories and statistical approaches [35]. Consequently, a combination of a HIDS and NIDS has been implemented to establish a hybrid IDS which can monitor network traffic and host activities

[35]. The advantages and disadvantage of both environments are listed in Table 2.

Table 2: Advantage and disadvantage of Host- and Network- IDS

Monitored environment	**Advantage**	**disadvantage**
Host-based IDS	- identifies the improper use of an organisation's internal equipment [33]	- can not be compatible to monitor different platforms, for example, API calls for Linux and DLLs for Windows operating systems [36]
	- is also used when the network payload was encrypted or obfuscated using metamorphic techniques or some evasion techniques such as fragmentation [33, 36]	- can be exposed as soon as its host server is compromised by an attacker [33] - is also not a good solution in the case of fast-spreading zero-day worms
Network-based IDS	- monitors network traffic over only a certain network segment regardless of the destination's type of operating system [16, 33] - can capture information from packet headers as well as packet payload if it does not encrypted	- can not easily handle scalable systems, as high-speed connected networks have become the norm of current networks [23]
	- is quite portable as it monitors network traffic over only a certain network segment [16]	- can not process encrypted data, as it can only capture information from packet headers [37]
	- can be installed on a network and its data are easily collected which is beneficial in some situations; for instance, following network topology [16]	

3.2. Detection methods

Intrusion detection methods are classified into four major types: Misuse-based (MDS); Anomaly-based (ADS); Stateful Protocol Analysis (SPA); and Hybrid-based (HDS) [6, 24]. A MDS monitors network traffic to match observed behaviours with attack signatures logged in a database. It produces higher detection rates and lower false alarm rates for known attacks than other types, but it cannot detect new or even variants of known attacks. This is a significant issue in terms of the computer security required to defend against those attacks. Moreover, a huge effort is necessary to repeatedly update its database that includes various rules for malicious activities, established by network security experts [38, 39]. To address some drawbacks of the MDS methods, Automatic Signature Generation (ASG) approaches have been proposed [40]. The approaches are broadly categorised into ASG without attack detection and

ASG with attack detection. The former does not use any attack detection methods prior to generating signatures such as Polygraph and Honeycomb system, whilst the latter identifies an attack vector, and then creates its signatures such as Honeycyber and Eudaemon systems.

An ADS creates a normal profile and identifies any variations from it as an suspicious event. It can identify known and zero-day attacks with less effort to construct its profile than a MDS, but it still faces some challenges presented in Section 8. A SPA examines protocol states, specifically a pair of request-response protocols, such as a HTTP protocol. Although a SPA is roughly similar to an ADS, it relies on vendor-developed profiles of certain protocols and requires information of the relevant network's protocol standard from international standard organisations [24]. As a SPA consumes many computer resources to inspect protocol states and is incompatible with different dedicated operating systems, an ADS is a better defence solution if its DE approach is properly designed [3, 41]. Finally, a HDS applies integrated methods to improve the detection accuracy. For example, MDS and ADS methods are accumulated for identifying certain known attack types and zero-day attacks, respectively [6, 23].

3.3. Applications and deployments

An IDS's deployment architecture is either distributed or centralised. The former is a compound system comprising multiple intrusion detection subsystems installed at different sites and connected to exchange relevant information. This helps in detecting malicious patterns which can identify corresponding attacks from multiple locations in a particular time. Conversely, a centralised IDS refers to a non-compound system which is deployed at only one site, with its architecture dependent on the organisation's size and sensitivity of the data which should be considered when designing a deployment [42]. IDSs are executed and installed to different applications and systems, as explained below.

- **Backbone-based IDSs** - are implemented on nodes of backbone, which is a portion of a network system that connects many network systems. A backbone IDS should monitor and analyse network data transmissions between different Local Area Networks (LANs) or sub-networks. A scalable NIDS server should be installed on a backbone network for monitoring all network traffic, and/or monitoring traffic for a specific server, gateway, switch or router. The multi-agent systems have been suggested for deploying NIDSs on backbone networks [43]. The design system could tackle the limitations of developing effective and efficient NIDS, but the important features and observations explained in Section 4.2 should be applied to enable running NIDSs in real-time. The individual agents do not capture the data from the network directly, but they receive the important features and observations. Every detection agent in the system uses misuse-based and/or anomaly-based detection methodology for recogninsing intrusive events. There are some challenges related to design adaptive multi-agent systems and scalable NIDSs that can handle large sizes and high speeds of current backbone networks. Designing a reliable IDS for high-speed backbone networks is still an open challenge, where the IDS should produce a low false alarm rate, especially for large-scale attack types such as DDoS [41].

- **Data center-based IDSs** - are deployed on key servers of a data center, which is a set of networked computers and storage that organisations utilise for processing, logging and disseminating large amounts of data. A data center-based IDS should inspect network packets that exchange between client-server and/or server-client systems. It should offer a solidified backup system and security management, as it monitors suspicious events of servers and devices with high bandwidth and a high-quality data flow control [44]. Since many companies have been using virtualisation technologies, migrating data centers and virtual machines is one of the biggest challenges in the cyber security domain. This is because new configurations and security tools used to track changes and monitor network systems have new vulnerabilities.

- **Access point-based IDSs** - are installed over access networks that link subscribers to a specific service provider, and across the carrier network, to other network systems (e.g., the Intranet and Internet). An access point IDS should identify abnormal activities through network systems that are connected by LANs and/or wireless LANs. Wireless Intrusion Detection Systems (WIDSs) have been proposed to monitor the radio spectrum of LANs and/or wireless LANs for identifying unauthorised access [45]. WIDSs are used to monitor and inspect traffic of sensors, servers and console. However, the heterogeneous sensors of antennas and radios that examine the wireless spectrum demand handling data dimensionality and developing self-adaptive NIDSs for defining malicious activities effectively.

- **Internet of Things (IoT)-based IDSs** - are deployed for protecting different applications based on the convergence of smart objects and the Internet [43]. An IoT-IDS should recognise anomalous behaviours from computers and physical devices, such as telemetry data of sensors and actuators, linked to the Internet. Traditional NIDSs have been used in IoT, but they generate large numbers of alerts involving high false alarm rates, due to overlapping legitimate and suspicious instances [17, 41, 43, 46]. Human network administrators cannot manually inspect these alerts to find attack observations [46]. Developing new post-processing techniques are essential in IoT networks for correlating NIDS alerts, reducing false alarm rates and visualising network data [41]. Moreover, the development of autonomic NIDSs with self-paradigm has become necessary in IoT. Based on this paradigm, new NIDSs could be configured, adapted and repaired, with low human interventions.

- **Cloud-based IDSs** - are deployed on nodes of centralised networks. A Cloud IDS is necessary for firms that migrate workloads and services to public Cloud paradigms, such as Amazon Web Services and Microsoft Azure for protecting models of platforms, software and infrastructures [18]. Existing NIDSs are not capable of detecting and responding to internal malicious activities and failing to protect Cloud computing and mobile Cloud computing. The detection of internal malicious activities is a challenging task, due to their potential complexity and remotely located modules [47]. Furthermore, many virtual machines could be deployed or destroyed at data centers of the Cloud, tracking normal and attack events demand scalable and collaborative IDSs.

- since many virtual machines are established and destroyed, the detection of attacks is a difficult task to monitor and track normal users and attackers over data centers.

- **Mobile edge computing/Fog-based IDSs** - are executed near to end users or networks' edges for protecting sensitive data that exchange over mobile, computer and network systems. when IDSs are deployed at the edges of networks, they would assist in addressing the Cloud challenges of processing large-scale networks, geographical distribution, high-mobility and low-latency. Processing network traffic between Cloud and edge sides is a main issue at the edge and Cloud paradigms, as they demand smart data management and scalable NIDS approaches that can efficiently recognise unknown suspicious events in real-time. The major challenges of edge computing are decentralised and distributed norm compared with the centralised norm of Cloud paradigms. The issues of the decentralised norm include the integration of different service infrastructures, and the need to synchronise soft and hard states of multi-tiered architecture. The issues of the distributed norm involve developing standards that specify how different elements of infrastructure providers can integrate with each other, and how virtual machines can access particular information such as context and host information [48].

New IDSs for the above applications should be capable of discovering known and zero-data attacks discussed in Section 2. Such systems should effectively and efficiently monitor high-speed networks that can exchange data at 10 Gbps or higher. Moreover, they should be scalable and self-adaptive for analysing diverse networks through wide areas in real-times.

3.4. Anomaly types

Anomalies are known as patterns in network traffic which behave differently from legitimate activities. Their types are classified as a point, contextual or collective according to the output from the detection method used [14, 23, 49]. A point anomaly occurs when a certain observation deviates from the legitimate profile and, in statistical methods, is referred to as an outlier. Contextual anomalies occur when data patterns are anomalous in a particular context and appear as related behaviours which are always different from the majority of normal activities. Collective anomalies happen when a group of similar data instances acts anomalously compared with the entire data of a normal network.

The output from anomaly detection is often based on a baseline/threshold which is a condition that discriminates between normal and attack instances [13]. Determining this threshold is one of the significant challenges faced when designing a NADS due to the overlapping patterns of normal and attack activities. The types of output from anomaly detection can be a score or binary which affects the selection of a correct threshold. A score-based output is a numeric value of either probabilities or real numbers for each data record while a binary/label-based output is a certain value which tags each record as normal or attack; for example, the labels of the KDD99 [50] and UNSW-NB15 [51] datasets are '0' and '1' for normal and attack records, respectively.

3.5. Defence responses

Defence/security responses are actions taken by a system against malicious events that are recognised. More specifically, they are the capability of identifying a given activity as an attack, and then a system administrator should take an action to stop the malicious activity [2, 23]. There are two types of responses: passive and active, as explained in the following.

- **Passive response**: is taken by a human administrator when an IDS identifies a malicious event. This process normally happens after gathering and correlating traces by the administrator, when an anomalous behaviour is detected and an alert is raised. The popular form of an alarm is a popup window or an onscreen alert. It can be displayed on the IDS console such as the snort alert console. There are SNMP traps and messages that create alerts and reports to the network management for taking actions.
- **Active response**: is also called an Intrusion Prevention System (IPS), which is an immediate and automatic action taken when malicious events are detected by executing a predefined script action. It allows to stop the progress of attacks by blocking their IP addresses and ports, changing the ACL, resetting the TCP protocol for terminating the connections, and/or re-configuring firewalls and routers.

4. Components of NADS

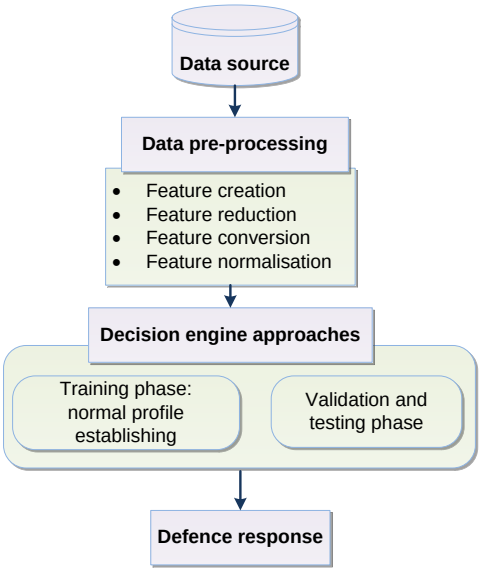

Figure 3: Components of NADS [2]

As depicted in Figure 3, a typical NADS consists of four components: a data source; data pre-processing module, DE method and security responses [52], as elaborated below. The factors involved in designing an effective NADS framework are encompassed by understanding its components.

4.1. Data source

The data source is a major component of any NIDS for evaluating the performances of DE methods, due to the difficulty of labelling legitimate and attack activities in live network traffic [53, 54]. Network data sources have been collected in a real-time data collection or off-line dataset which comprises a wide variety of normal and malicious records.

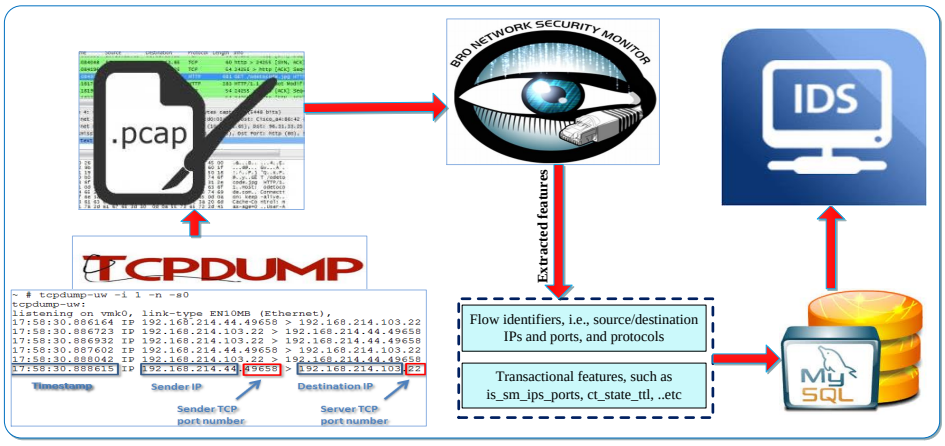

Figure 4: Process of sniffing and creating network features in real-time

With the high speeds and large sizes of current network environments, network data has the characteristics of big data which is typically defined in terms of volume (i.e., the amount of data), velocity (i.e., the speed of data processing) and variety (i.e., the complexity of the data and to what extent they are of diverse types and dimensions) [55]. As traditional database systems generally cannot process the big data contained in real-world problems, it is vital to use, for example, the Hadoop [56] or MySQL Cluster CGE [57] tools to store and handle a network's big data as a data management unit for NIDS technologies [2, 3].

In real-time processing, network traffic is collected to monitor and detect abnormal activities. Bidirectional or unidirectional network flows are aggregated at the choke-points, for example, ingress router and switch devices, to reduce the network's overheads. These devices have limited buffers and simple mechanisms for collecting flows which can accumulate using only one attribute for a given time, such as source/destination IP addresses or protocols. To address this limitation, the simple random sampling technique is basically applied to select data portions each time. The technique randomly chooses a sample of a given data size that no observations are included more than once, with all subsets of the observations given an equal probability of selection [58].

To give an example of extracting network features, many tools such as tcpdump, Bro-IDS and MySQL Cluster CGE are utilised as shown in Figure 4. The tcpdump tool is applied to sniff network packets in the format of pcap files. After that, the Bro-IDS is used for extracting the flow-based features and general information about different protocol types from the pcap files. The extracted features are stored in a MySQL database to make it easier to create labelling

the vectors, either normal or abnormal. Finally, in the IDS, a DE approach is used for discovering existing and zero-day attacks from the features.

In order to design an effective NIDS, there are several offline datasets as data sources generated for training and validating NIDSs. The existing datasets could be applied to different IDS-based applications discussed in section 3.3 while anlysing computer and network systems. We classify the benchmark datasets into old and new ones as follows, and their comparisons are listed in Table 3.

Table 3: Comparisons of popular datasets

Datasets	**Realistic network configuration**	**Realistic network traffic**	**Labelled observations**	**Total interaction capture**	**Full packet capture**	**Many malicious scenarios**
KDD99 and NSL-KDD	*True*	*False*	*True*	*True*	*True*	*True*[8]
CAIDA	*True*[1]	*True*	*False*	*False*[5]	*False*[4]	*False*[2]
DEFCON	*False*	*False*[5]	*False*	*True*	*True*	*True*[8]
LBNL	*False*	*True*[1]	*False*	False	*False*[4]	True
UNIBS	True	*True*[2]	*True*	*True*[4]	*True*	*False*[2]
TUIDS	*True*	*True*[9]	*True*	*True*[4]	*True*	*True*[8]
ISCX and CICDS2017	*True*[2]	*True*[6]	*True*	*True*	*True*	*True*
DARPA-2009	*True*[1]	*True*[5]	*False*	*False*[5]	*True*	*True*
UNSW-NB15	*True*	*True*[9]	*True*	*True*	*True*	*True*[10]
NGIDS-DS	*True*	*True*	*True*	*True*[3]	*False*	*True*[10]

1. Network configuration information not available
2. Basic captured network traces
3. No payload available; most simply reduced/summarised trace information
4. No payload available; in some packets, protocol, destination and flags deleted
5. Comprises no packet contents and no host or protocol information
6. Designed to include profiles of network information
7. Only malicious traffic
8. Does not reflect current trends
9. Contains a large number of protocols and services
10. Has modern security events and malware scenarios

Old datasets

- **The KDD99 and NSL-KDD datasets** - the IST group at the Lincoln Laboratories in the MIT University performed a simulation involving both normal and abnormal traffic in the military network of the U.S. Air Force LAN environment to generate the DARPA 98 dataset using nine weeks of raw tcpdump files [50]. The NSL-KDD dataset [59] is an enhanced version of the KDD99 dataset. This dataset tackles some drawbacks of the KDD99 dataset. Firstly, it does not contain duplicated observations in either the training or testing set. Secondly, the numbers of observations in the training and testing sets are adopted from different portions of the original KDD99 dataset without any duplication. Nevertheless, the KDD99 and NSL-KDD datasets cannot represent contemporary network

traffic as its legitimate and attack behaviours are extremely different from those of current network traffic.

- **The CAIDA datasets** [60] are collections of different data types for analysing malicious events to validate attack detection approaches, but are limited to particular types of attacks, such as DDoS ones, with their traces the anonymised backbones of the packet headers without their payloads. The most common CAIDA dataset is the CAIDA DDoS 2007 anomaly one which includes an hour of anonymised network traffic for DDoS attacks. These datasets did not have a ground truth about the attack activities involved and, moreover, their pcap files were not inspected precisely to elicit features in order to discriminate attack activities from normal ones.

- **The DEFCON dataset** is freely available on the internet [61]. Although most of the files are full packet capture ones, some have truncated frames. They were extracted during a hacking competition named capture-the-flag in which competing teams were divided into two groups: hackers and defenders. It contains only malicious activities with no legitimate traffic which is different from realistic network environments. This dataset is only effective for assessing alert correlation approaches and poor for evaluating NADS ones due to its limitations of losing frames and lacking legitimate network traffic.

- **The UNIBS dataset** [62] was gathered from the network router of the University of Brescia, Italy, on three days. Its traffic was collected from 20 workstations running the GT client daemon using the tcpdump tool. The raw packets were captured and logged on a disk of a workstation linked to the router across an ATA controller.

- **The LBNL dataset [63]** was designed at the Lawrence Berkeley National Laboratory (LBNL) that includes header network traces without payload. The dataset was anonymised for excluding any sensitive information which could recognsie individual IP addresses. Its network packets were collected from two routers at the LBNL network that includes about thousand host systems for nearly hundred hours.

- **The Kyoto dataset** developed at Kyoto University, is a set of network traffic collected from honeypot systems. It was created using the BRO-IDS tool to extract 24 features from the KDD99 dataset which were then categorised into 14 conventional and 10 additional features that reflected the network's characteristics [64]. However, its main drawbacks are that it lacks measures for labelling and describing attack behaviours or even variants of legitimate ones.

- **The DARPA 2009 dataset** [65] was synthetically designed to emulate the traffic between 16 sub-networks and the internet with data collected over 10 days, from 3rd to 12th November 2009. It contains synthetic HTTP, SMTP and DNS background data traffic, and has a set of attack types such as DoS and DDoS. It consists of 7000 pcap files with almost 6.5 TB, with each file including approximately a one- or two-minute timing window.

- **The CDX dataset** [66] was synthetically developed by the Cyber Research Center at the US Military Academy. It associates IP addresses found in PCAP files with IP addresses of clients on the internal USMA network. It was created during a network warfare competition for the design of a tagged dataset. It comprises ASNM features generated from the tcpdump capture of malicious and normal TCP communications on network services which are vulnerable to DoS attacks.

- **The CTU-13 dataset** [67] which was developed at the CTU University, consists of a collection of a large number of botnets and normal traffic involving 13 captures of different botnet scenarios. In each scenario, a particular malware, which used many protocols and executed different actions, was implemented.

New datasets

- **The ISCX dataset** [68, 69] was designed using the concept of profiles which contains descriptions of attacks and distribution models for a network architecture. Its records were captured from a real-time simulation conducted over seven days of normal network traffic and synthetic attack simulators. Several multi-stage attack scenarios were included to help in evaluating NIDS methods. However, the dataset did not provide the ground truth about attacks to reflect the credibility of labelling and, secondly, the profile concept used to build the dataset could be impossible to apply in a real complex network because of the difficulty of analysing and logging.

- **The TUIDS dataset** [70] was collected from the Network Security Lab at the University of Tezpur, India based on different attack scenarios. Its network packets were captured using the nfdump and gulp tools for capturing representative features. Their features are categorised into basic, content, time, window and connectionless, with adding their labels either normal or attack.

- **The ADFA dataset** [71] was developed at the University of New South Wales to evaluate Linux and Windows HIDSs. It contains host logs that were manually designed using different simulation configurations. The Linux data collection includes system call traces generated by the Linux auditd program and then processed by size. For the training set, traces larger than 300 bytes to 6 kB and, for the validation set, those outside the range of 300 bytes to 10 kB were neglected. Windows XP was used to generate a set of DLL calls of 1828 normal and 5773 attack traces.

- **The UNSW-NB15 dataset** [51, 72] was developed at the University of New South Wales for evaluating new NIDSs. It has a large collection of authentic recent legitimate and anomalous vectors. The size of its network packets is about 100 Gigabytes extracted 2,540,044 vectors and are stored in four CSV files. Each vector consists of 47 features and the class label. Its speed is in average of 5-10 Megabytes per second between source and destination IP addresses. It comprises ten different classes, one normal and nine types of attacks.

- **The CICIDS2017 dataset [73]** was generated at the Canadian Institute for Cybersecurity. It involves recent normal and attack scenarios using the concept of data profiling like the ISCX dataset. Its traffic was analysed using the CICFlowMeter with tagged flows using the time stamp, the source and destination ports and IP addresses, and protocol types.
- **The NGIDS-DS dataset** [74] was designed at the University of New South Wales for assessing Linux HIDSs. The network packets between the attacking system and victim system were captured in one pcap file. It consists a huge number of normal and abnormal vectors generated using the IXIA perfect-storm tool saved in several CSV files. System calls and their execution times where captured from the victim Linux operating system as feature sets that will be used to evaluate the efficiency of new HIDSs.

4.2. Data pre-processing

Data pre-processing is a significant step in learning theories because, like data-gathering measures which are often loosely controlled and result in irrelevant or duplicated data values, network data extracted from network traffic also include these data. It filters network data by removing redundant, noisy or irrelevant information which leads to improving the performance of DE approaches for detecting attack behaviours. Data pre-processing for network data involves the creation, reduction, conversion and normalisation of feature, as described in the following.

4.2.1. Feature creation

Network features are captured from raw network packets using different tools, such as Argus, BRO-IDS, Netflow, Tcptrace and Netmate. A NIDS requires a set of features such as the features, as in the KDD99 and UNSW-NB15 datasets. Moreover, additional features are established using both transactional flow identifiers (i.e., source and destination IP addresses) and transactional connection times (e.g., 10 or 100 connections per second) to define the potential characteristics of network behaviours [4, 72, 75]. These features are significant for identifying attackers who scan victims in a capricious way, such as one scan per minute or per hour; for example, in the KDD99 and UNSW-NB15 datasets, the *is_sm_flw* feature could identify land or teardrop attacks [76].

The potential process of sniffing and creating the features of the datasets are demonstrated in Table 4. For creating the features from the datasets, a sniffer module such as the tcpdump tool was utilised to capture network traffic in the pcap format. After that, a network extractor module was applied such as snort or BRO-IDS to extract important features from the seven layers of the OSI model. It is very essential to select only significant features that can discriminate between normal and abnormal observations by using DE approaches [4, 5, 51, 72].

4.2.2. Feature Reduction (FR)

It is the method of removing unimportant/noisy features, and can be separated into feature selection and feature extraction. The former finds a subset of the original features and the latter transforms the data from a high- into low-dimensional space [4, 77, 78]. FR is used in the data pre-processing component

Table 4: Creation of features of popular datasets

Datasets	Feature created
KDD99, NSLKDD and Kyoto	The BRO-IDS tool was used to extract a set of features from the tcpdump files of these datasets
CADIA, DEFECON LBNL and UNIBS	Only pcap (tcpdump) formats, without features generated from their traces, are available
TUIDS	The gulp and nfdump tools were utilised for generating a set of flow- and packet-level features
ISCX and CICDS2017	The Snort, QRadar, OSSIM IDS management systems and ntop visualisation systems were used to generate attributes from different protocols and services
CDX	The Snort IDS system generated a set of rules for use as features
DARPA-2009	The Tcptrace tool was used to extract features from the pcap files
ADFA and NGIDS-DS	The Linux auditd tool was applied to generate system call identifiers from Linux hosts
UNSW-NB15	The Bro-IDS and Argus tools, and extractor module were used to extract different features from the pcap files

for building an effective NADS in which it plays a significant role in efficiently and effectively detecting network attacks. As network packets have some information which might be important for identifying anomalies, they should be carefully analysed to select only the relevant information that can help a DE approach correctly detect anomalous activities. Feature selection methods comprise four steps, subset generation, subset evaluation, a stopping criterion and result validation, as depicted in Figure 5 (see [23, 79]).

- **Subset generation** - is an fundamental heuristic search step whereby each state in the search space specifies a candidate subset for the assessment step. For a dataset with D features, there are 2^D candidate subsets, a space that enables an excessively thorough search with even only a reasonable number of features [80].
- **Subset evaluation** - each new subset created has to be assessed using an evaluation measure which can be classified as either independent or dependent based on the learning techniques in which it is applied on the selected features [80].
- **Stopping criterion** - controls when a FS method should end, with common ones the minimum number of features selected, maximum number of iterations and completion of the search [80].
- **Result validation** - a simple means of validating results is estimating the output using prior information about the data [23, 80].

Popular techniques for reducing network features. The Association Rule Mining (ARM) [81], Principal Component Analysis (PCA) [82] and Independent Com-

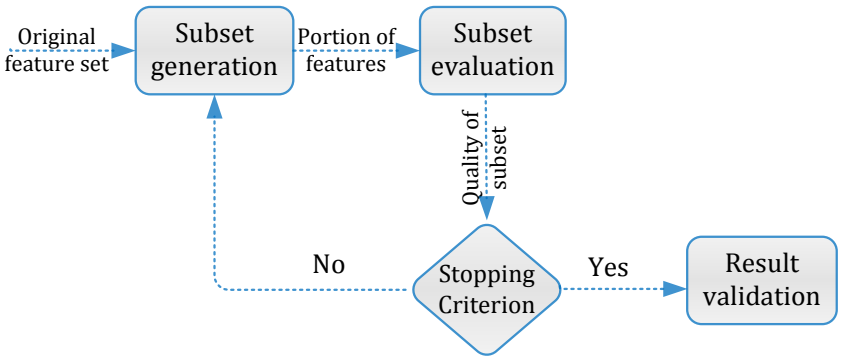

Figure 5: Main steps in feature selection

ponent Analysis (ICA) [83] techniques are widely used for selecting important network features, as described in the following.

- **ARM** - is a data mining technique used to compute the correlation between two or more variables in a dataset by determining the strongest rules that occur between their values.
- **PCA** - sorts a set of attributes based on the highest variations for each attribute and generates a new dimensional space of uncorrelated attributes by omitting those with low variances.
- **ICA** - is a generative model which generalises the PCA technique. It mines unidentified hidden components from multivariate data, that is, linear mixtures of some hidden variables, using only the assumption that the unknown components are mutually independent and non-normal distributions.

Many studies [77, 78, 84, 85] have used the ARM technique in a NADS to detect abnormal instances. Luo et al. [86] used the ARM to construct a set of rules from audit data to establish a normal profile and detect any variation from it as an attack. Yanyan and Yuan [87] developed a partition-based ARM technique for scanning the training set twice. In the first scan, the data is divided into many partitions to run easily in memory while, in the second, itemsets of the training set are created.

As several research studies have been undertaken using the ICA and PCA techniques to analyse the potential properties of network traffic and eliminate inappropriate or noisy features, these mechanisms are usually utilised in the data pre-processing module to address the variety problem of big data discussed in [88, 89]. In [90], a NADS technique using the ICA mechanism was developed to detect stealthy attacks with a high detection accuracy. It was assumed that the hacker has no information about the system, and malicious activities were detected based on a measurement matrix. De la Hoz et al. [91] suggested an adaptive IDS based on a hybrid statistical technique using PCA, the fisher discrimination ratio and probabilistic self-organising maps (SOMs).

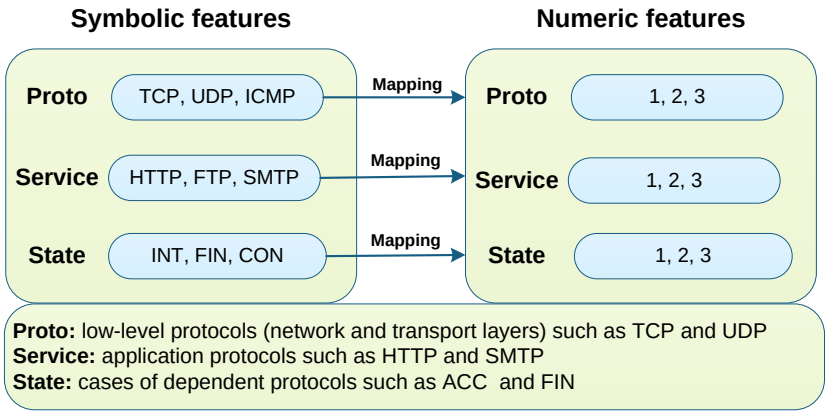

Figure 6: Example of feature conversion using UNSW-NB15 dataset

4.2.3. Feature conversion

NIDS datasets include quantitative (i.e., numeric) and qualitative (i.e., symbolic) features. Since a statistical DE can deal with only quantitative data, a unified format for features (F) is used to map symbolic features into numeric features (i.e., $F \in R$), where R indicates real numbers [3, 75]. In other words, symbolic data are replaced with sequential numbers for ease of processing in statistical approaches. As shown in Figure 6, we provide an example of converting three symbolic features into numeric ones using the UNSW-NB15 dataset.

4.2.4. Feature normalisation

This is a function for scaling the feature's value into a specific confidence interval, such as [0, 1] [2, 75]. Its main benefit is to remove the bias from raw data without amending the statistical characteristics of the features. Common functions of normalisation are the linear transformation and z-score, as given in equations (1) and (2), respectively.

$$X_{normalised} = (X - min(X))/(max(X) - min(X)) \tag{1}$$

$$Z = (X - \mu)/\sigma \tag{2}$$

where X denotes the feature values, μ is the mean of the feature values and σ is the standard deviation.

For example, Table 5 lists an example of feature normalisation, where three features with five rows from the UNSW-NB15 data were normalised using equation (1).

5. Decision engine (DE) approaches

The DE module of a NADS is clearly a critical aspect in the design of an efficient system for discovering intrusive activities in real time. DE approaches are

Table 5: Example of feature normalisation using UNSW-NB15 dataset

Original data			Normalised data		
Sload	Ct_srv_dst	Sinpkt	Sload	Ct_srv_dst	Sinpkt
14158.942	1	24.296	0.965	0.580	0.409
8395.112	6	49.915	0.690	0.555	0.411
1572.272	6	231.876	0.806	0.549	0.394
2740.179	1	152.877	0.813	0.599	0.787
8561.499	39	47.750	0.808	0.227	0.398

Sload: source bits per second
Ct_srv_dst: a number of connections containing the same service and destination address for each 100 flows
Sinpkt: source inter-packet arrival time (ms)

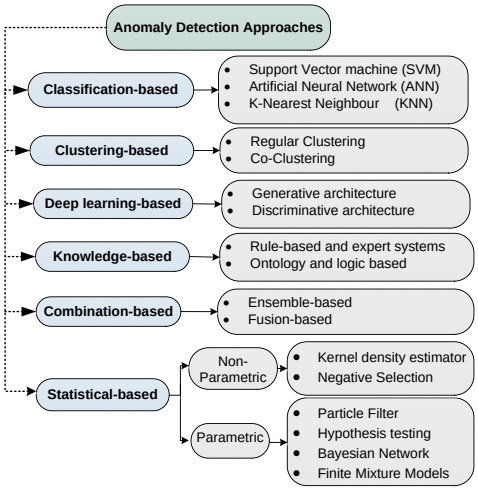

Figure 7: Taxonomy of network anomaly detection approaches

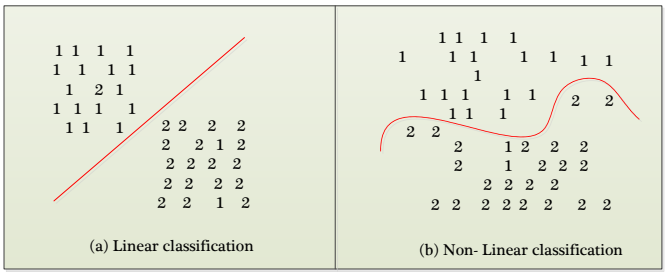

Figure 8: Classification types

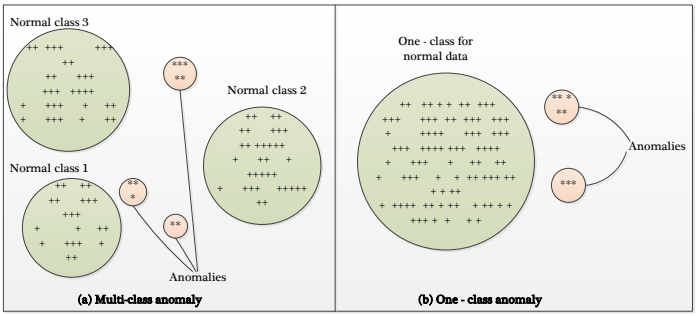

Figure 9: NADS classifications [23]

classified in six categories, classification-, clustering-, deep learning-, knowledge-, combination- and statistical-based [2, 14, 20, 23], as depicted in Figure 7, and explained as follows.

5.1. Classification-based approaches

Classification is categorising data instances in certain classes based on those in a training set while a testing set contains other instances for validating the labelling process; for example, assuming that we have two classes in which observations are labelled '1' and '2', these observations can be classified as linear or non-linear, as depicted in Figure 8. Classification approaches have been used to build models that enable classifying network traffic behaviours into either two classes (i.e., normal or attack) or a set of classes (i.e., normal with each attack as a class) [23, 92, 93], as depicted in Figure 9.

One-class anomaly methods become more interesting when there is an imbalance between the numbers of normal and attack observations, where those of normal instances are considerably greater than those of attack or rare events which is the nature of network traffic. They are also significant if instances are classified as normal or attack without any attack types, such as DoS and DDoS, being detected. Conversely, multi-class anomaly methods are more important if there is a balance between the classes of normal and attack observations, and, moreover, preferable for recognising attack types.

Discriminatory methods cannot be used to their full potential in such situations since, by their very natures, they rely on data from all classes to build the discriminatory functions that differentiate among the various classes. As a result, one-class learning methods, which use data from only a single class to build a model for recognising data from that class and rejecting the rest, have become more appealing.

The most popular classification-based techniques applied for NADSs are the Support Vector Machine (SVM), K-nearest Neighbour (KNN), as well as shallow and deep Artificial Neural Network (ANN). A typical SVM involves two steps for classifying data observations [94]; firstly, the training set is moved from the original input space into a higher-dimensional feature space based on kernel functions to convert a linear non-separable problem into a linearly separable one; secondly, the data points are on a hyperplane with the maximal margins at the nearest data points on each side. A one-class SVM [95] uses only the training set of legitimate network data and considers any deviation from the normal patterns as an anomaly. Wagner et al. [88] used a one-class SVM technique to establish a NIDS approach which detected zero-day attacks that did not belong to the normal training class. However, this technique often took a long time to train a large amount of data, such as network data. Similarly, Horng et al. [96] proposed a NADS which included a hierarchical clustering and SVM to decrease the processing time of the training phase and enhance detection rate. In [97], a least-square SVM was proposed for the design of a lightweight NADS by selecting the significant features of network data and detecting anomalies.

A KNN mechanism classifies each observation assigned to the class label by computing the highest confidence between the k data points nearest the query data point [52]. A KNN-based NADS creates a normal network profile and treats any deviation from it as an attack. It is a powerful DE for NADSs because it does not demand adapting parameters in the training stage. The KNN technique was used to design a Dependable NIDS (DIDS) based on the strangeness and isolation measures of its potential functions which could effectively identify network attacks [98]. Nevertheless, KNNs are often time-consuming and require vast amounts of storage to classify high-speed network traffic.

Other classification techniques, for instance, a decision tree, regression models and fuzzy logic (see [12, 13, 23]) have also been applied to design NADSs. However, overall, classification-based IDSs rely heavily on the assumption that each classifier has to be adjusted separately and always consume more resources than statistical techniques. Ultimately, if these techniques do not successfully build normal patterns, they are not capable of detecting new attacks. It is important to note that most classification techniques have been evaluated using old datasets, particularly the KDD99 dataset, and their poor performances will certainly be worse on newer datasets.

5.2. Clustering-based approaches

Clustering approaches are unsupervised machine-learning mechanisms which assign a set of data points to groups based on the similar characteristics of these points, such as distance or probability measures; for example, if we have unlabelled data instances in two dimensions (X and Y), we might group them into four clusters, namely C_1 to C_4, as shown in Figure 10 (a). Another concept derived from clustering is outliers which denote that some data points in a dataset more highly deviate than regularly grouped ones; for example, in Figure

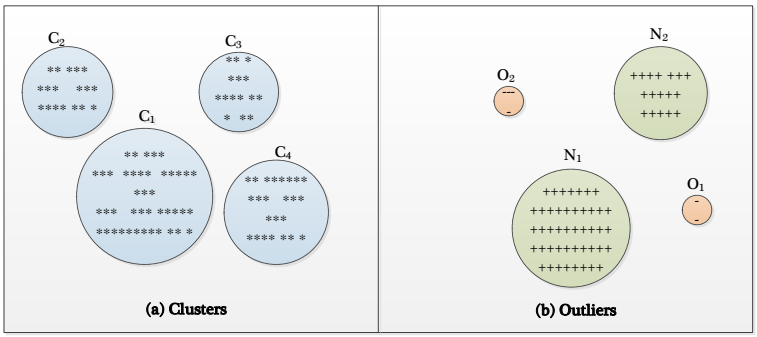

Figure 10: Methodologies of clusters and outliers [23]

10 (b), the data points of O_1 and O_2 are outliers while those of N_1 and N_2 are normal clusters [99].

Although there are different clustering techniques, the most popular types applied for NADSs are regular and co-clustering with the difference between their strategies of processing the observations and features of a network dataset [13, 14, 23]. Specifically, regular clustering, such as K-means clustering, assembles data points from the observations of a dataset while co-clustering simultaneously considers both the observations and features of a dataset to provide clusters.

When using clustering to identify anomalies, three key assumptions are usually made. The first is that, as legitimate data instances often fall into a cluster whereas attacks do not, in a NADS methodology, clustering identifies any data instances that do not fall into a legitimate cluster as attacks, with noise data also considered anomalous, as in [100]. A drawback of this assumption is that clustering techniques cannot be optimised to identify anomalies as the major goal of a clustering algorithm is to define clusters. Secondly, legitimate data instances are usually located near the closest cluster centroid while anomaly ones are often far away from it [13].

Techniques using this assumption consider the points farthest from the cluster centre as anomalies, with many of them suggested for designing NADSs [13] whereas, if anomalies are located in normal clusters, they cannot be correctly identified. To tackle this challenge, the third assumption is that legitimate data instances fall into vast and dense clusters and anomalies into small or spare ones. Mechanisms using this assumption identify data observations belonging to clusters with those of sizes and/or densities under a baseline considered anomalies.

Bhuyan et al. [101] designed an outlier-based NADS in networks in which legitimate data were clustered using a k-means technique and then a reference point computed for each cluster, with these points classified as attacks if they were less than a certain threshold value. Also, in [102], a NADS for large network datasets using tree-based clustering and ensemble-based techniques for improving accuracy in a real network environment was proposed. Nadiammai et al. [103] analysed and evaluated k means, hierarchical and fuzzy c-means clustering techniques for building a NADS. However, this system could not work effectively on an unbalanced data problem in which the network instances

of normal class are too larger than the instances of abnormal class.

Clustering-based NADS techniques have several advantages. Firstly, they group data points in an unsupervised manner which shows that they do not need to provide class labels for observations, which is a very difficult process, to ensure the correct labelling of data as either normal or attack. Secondly, they are effective for clustering large datasets into similar groups to detect network anomalies, which decrease computational complexity, and perform better than classification methods. In contrast, one of clustering-based NADS drawbacks is that its clustering is highly reliant on its efficacy in profiling normal instances while another is that dynamically updating a profile for legitimate network data is time-consuming. Finally, its dependency on one of the three above assumptions is occasionally problematic for effectively recognising abnormal behaviours as it produces a high false alarm rate and, in particular, attack instances can conceal themselves in a normal cluster.

5.3. Deep Learning- based approaches

The foundation theory behind shallow and deep learning methods is the utilisation of advanced ANN architecture that is inspired by the human brain and compute an entirely different way than traditional digital methods. ANNs are machine learning algorithms which convert the inputs into outputs through non-linear latent processing of a set of artificial neurons, and these methods are classified into shallow and deep learning [104]. A shallow network is an ANN which contains often one/two hidden layer(s), whereas a deep network consists of multiple hidden layers with several architectures, as depicted in Figure 11.

Recently, deep learning networks are widely used for various pattern recognition and network applications, due to their capability of learning a computational process in depth. In a NADS methodology, shallow and deep networks require some information about the legitimate data class to systematically alter the interconnection neurons to learn the weights of the network and obtain a model that can discriminate attacks from normal behaviours. Deep learning networks are classified into different types relying on its architectural design that comprise hierarchical layers of non-linear processing levels [105]. Based on Hodo et al. [106], deep networks are categorised into generative and discriminative architectures. The generative architecture computes joint probability distributions from observed data with their classes, which involves the following models.

- **Recurrent Neural Network (RNN)** - is a supervised and/or unsupervised learning model. The core theory behind RNN is that information is connected in long sequences via a layer-by-layer connection with a feedback loop. There is a directed cycle between its layers that increase its reliability, with the capability of creating an internal memory for logging data of the previous input.

- **Deep Auto Encoder (DAE)** - is used for learning efficient coding in an unsupervised manner. The simplest architecture of DAE involves an input layer, more than one hidden layer and an output layer that has the same number of neurons in the input layer for reconstruction.

- **Deep Boltzmann Machine (DBM)** - is an indirect probabilistic model that includes energy and stochastic units for the overall network to pro-

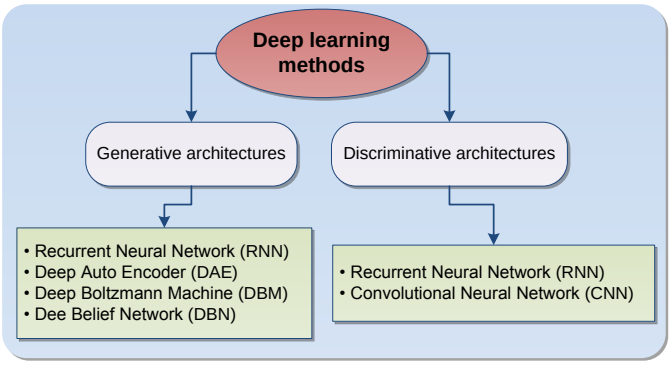

Figure 11: Methods of deep learning

duce binary results. A Restricted Boltzmann Machine (RBM) is applied to reduce hidden layers, which does not allow intra-layer connections between hidden units. Training a stack of DBM using unlabeled data as the input of the next layer and inserting a layer for discrimination could lead to building an architecture of DBN.

- **Dee Belief Network (DBN)** - comprises many hidden layers, where a connection is between layers not between units within each layer. It is a collection of unsupervised and supervised learning networks. The unsupervised model is learned by a greedy layer-by-layer connection at a time, whereas the supervised network is one or more layers connected for classifying tasks.

The discriminative architecture estimates posterior distributions of classes conditioned on the observed data that comprises RNN and Convolutional Neural Network (CNN), discussed below.

- **RNN**- uses discriminative power for a classification task, and this occurs when the output of the model is labelled in a sequence with the input.
- **Convolutional Neural Network (CNN)** - is a space invariant multi-perceptron ANN, which is biologically inspired by the organisation of the animal visual cortex. It has many hidden layers, which typically consists of convolutional layers, pooling layers, fully connected layers and normalisation layer. The convolutional layers share many weights that have small parameters, making the CNN easier in the training process compared to other models with the same number of hidden layers.

Multiple research studies [105, 106, 107, 108, 109, 110] have recently applied deep learning techniques to NADSs. Alom et al. [107] used a DBN-based NADS by configuring a greedy layer-by-layer learning algorithm to learn each stack of RBM at a time for discovering intrusion events. In [108], a deep auto-encoder technique was developed to reduce data dimensions that was considered a pre-stage for classifying network observations. A shallow ANN algorithm was applied as a classifier to assess the effectiveness of an auto-encoder technique

compared with the PCA and factor analysis algorithms. Yin et al. [109] proposed RNN-based NADS IDS for recognising malicious network instances. The experiments were conducted on differnt hidden nodes and learning rate values.

In [110], the author proposed an ensemble method-based NADS that involves DFN architectures that contain shallow auto-encoder and DFN, DBN and DNN architectures. The method was assessed using the NSL-KDD dataset, and the experiment results showed a reasonable performance for discovering abnormal network activities. It is observed that deep learning algorithms could considerably enhance the NADSs' performance, with high detection accuracy and low false alarm rates. However, they usually consume a long time to process a network data to determine the best neural weights for minimising classification errors as possible.

5.4. Knowledge-based approaches

Knowledge-based techniques establish a set of patterns from input data to classify data points with respect to class labels. In MDSs, network traffic data are examined against predefined patterns of attacks and system vulnerabilities to detect malicious events and raise an alarm [111]. Although these approaches can identify known attacks, they cannot determine zero-day ones unless a profile is constructed from diverse normal patterns as NADSs [12].

Common knowledge-based NADS approaches are rule-based and expert as well as ontology- and logic-based [23]. Rule-based methods model the knowledge collected about suspicious network events which allows browsing of network traffic data to find evidence of existing vulnerabilities [112]. An expert system comprises rules which define attack events whereby network traffic data are transformed into patterns according to their relative weights in the system and an inference engine matches the predefined rules with the current state of the system to detect attack activities [113]. Rule-based and expert system approaches have been widely applied to detect suspicious network events while ontology- and logic-based ones model intrusion signatures based on a logic structure by incorporating the constraints and statistical characteristics of network traffic data [23].

Snort [114] is one of the popular rule-based and open-source IDSs. Its rules recognise malicious network packets by matching the current packet against predefined rules and cannot detect zero-day attacks but produce a high FPR due to its methodology for identifying attack signatures [115]. Currently, Snort involves more than 20,000 rules which are usually updated by users [12]. The Petri nets tool [116, 117], which was designed at Purdue University, is an example of a knowledge-based IDS which consists of directed bipartite graphs and Coloured Petri Nets (CPNs) representing the signatures of intrusions. This tool was used for developing the Intrusion Detection in Our Time (IDIOT) tool for detecting misuse events [116]. Although this tool can easily represent small network data and helps in discriminating known attacks, its process for matching an attack signature with predefined rules is very difficult to execute in real network environments and takes a long processing time. Vaccaro et al. [118] proposed an intrusion-detection tool which identifies malicious statistical behaviours by establishing a set of rules that statistically depicts the behaviours of users using logs of their activities over a certain period of time. It then matches the current activity against the stored rules to detect suspicious behaviours.

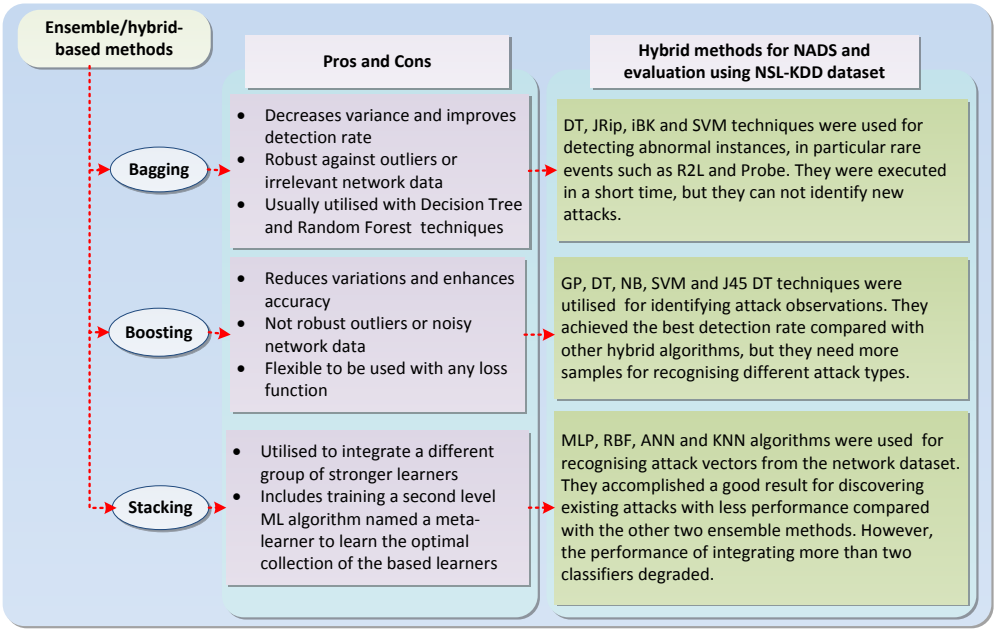

Figure 12: Comparison of ensemble-based methods used for NADS

Naldurg et al. [119] suggested an intrusion detection framework using temporal logic specifications with attack patterns formulated in a logic structure called EAGLE. It supported data values and parameters in recursive equations and enabled the identification of intrusions with temporal patterns. Hung et al. [120] presented an ontology-based approach for establishing a NADS according to the end-users' domain in which, as ontologies were applied as a conceptual modelling technique, a NADS could be simply built.

Knowledge-based NADS mechanisms have some advantages: firstly, they are sufficiently robust and flexible to discriminate existing attacks in small network traffic data; and, secondly, achieve a high detection rate if a significant knowledge base about legitimate and anomalous instances can be extracted correctly. On the contrary, they have FPRs due to the unavailability of biased normal and intrusion network traffic data and cannot identify rare or zero-day anomalies. Finally, their procedures for dynamically updating rules are very challenging and their processing times very expensive which are deterrents to building an online NADS [23].

5.5. Combination-based approaches

A combination-based methodology uses multiple mechanisms to classify data points effectively and efficiently, with most of those used for NADSs ensemble- and fusion-based mechanisms, as shown in Figure 12. Ensemble learning approaches integrate many techniques and consolidate them to achieve an overall accuracy which outperforms that of each classifier [23, 34, 121, 122] and are categorised as bagging, boosting and stack generalisation/stacking [15, 23]. Firstly, bagging, so-called bootstrap aggregation, improves the detection accuracy by

establishing an enhanced composite classifier which combines the findings of previously used classification techniques into one predictor. Secondly, boosting constructs an incremental ensemble by learning misclassified observations acquired from a previous model. Thirdly, stack generalisation obtains the highest generalised accuracy by utilising the probabilities for each class from a particular classification algorithm.

Fusion-based approaches, which integrate the decisions coming from different classifiers, have emerged as techniques that could reinforce the final decision [123], with their taxonomy consisting of three levels, data, feature and decision. Some methods tackle the problem of high dimensionality by adopting only relevant attributes while others amalgamate classification techniques trained on diverse attributes using either hierarchical abstraction levels or the types of attributes involved [23].

Ensemble- and hybrid-based methods have been applied to design effective NADSs. The Random Forest technique is one of the popular ensemble approaches compounded by decision trees. Its output contain the mean of the leaves for the regressive aspect or the majority vote for the classification aspect. More details of using Random Forest based NADSs are provided in [20]. Folino et al. [124] provided a distributed data mining technique for improving the accuracy of intrusion detection based on genetic programming extended with ensemble learning. Perdisci et al. [125] established a payload NADS based on a hybrid of one-class SVM techniques for improving the accuracy of detection. Nguyen et al. [126] suggested a classification technique using both the input features and additional ones provided by k-means clustering. These ensemble methods were computed using the classification capabilities of techniques for different local data segments provided by k-means clustering. Aburomman et al. [127] suggested an ensemble method which used PSO-generated weights to build a hybrid of more accurate classifiers for NADS created based on local unimodal sampling and weighted majority algorithm approaches to enhance the accuracy of detection rate.

Combination-based techniques are advantageous as they achieve higher accuracy and detection rates than single ones while requiring some parameters that can be precisely adjusted. However, adopting a subset of consistent and unbiased classification techniques is difficult because it depends on usinng a hybridisation measure to combine them. Also, it is evident that their computational costs for huge amounts of network packets are high because of the number of classifiers used [2, 23, 41].

5.6. Statistical-based approaches

From the statistical aspect, an anomaly is a rare event which occurs amongst natural data events and is measured by statistical approaches which could be of the first order, such as means and standard deviations, the second order, such as correlation measures, or the third order, such as hypothesis testing, mixture models and inference approaches. In NADSs, these approaches fit a statistical model of legitimate network data and then utilise a statistical test, using either a threshold/baseline or probability condition, to identify deviated instances as anomalies [128]. Statistical-based approaches are classified as non-parametric and parametric [23, 128, 129], both of which have been widely applied to develop statistical models for NADS.

5.6.1. Non-parametric approaches

The approaches do not make any assumptions about the statistical characteristics of given data. They create a model as they run and attempt to resolve the complexity of the data to efficiently adapt the data points. One of the simplest non-parametric statistical approaches is using histogram tools which graphically illustrate the tabulated frequencies of data [53]. In a NADS, a normal histogram is built and then new tested data points determined which, if they do not fall into the normal histogram are considered anomalous instances. For multivariate network data, feature-level histograms are established, with an overall score for a test data point attained by accumulating the scores of selected features.

The methodologies of the most commonly used non-parametric methods are as follows.

- **Kernel density estimator**

  The kernel density estimator is a non-parametric method that bases its estimations on some kernel distributions, such as Gaussian, for all the sample space data and then integrates the local contributions of all the distributions [130]. Estimating the probability density of each sample depends on the data points that fall in a localised neighbourhood of the kernel. For instance, Shen et al. [130] suggested a NADS based on a non-parametric method which simulates the PDFs of some random variables. A set of kernel density estimators was established and the distribution parameters estimated to classify malicious and normal instances. This method was extended in [131] to build a non-stationary high-dimensional PDF estimator using parallel programming to identify computer intrusions.

- **Negative selection**

  Negative Selection (NS) techniques have been widely applied for detecting anomalous network instances. The theory behind NS was inspired by the characteristics of the human immune system which can identify antigens [132], meaning that anything that is not a portion of the human body can be detected, for example, viruses and bacteria. Attack detection has the essential objective of differentiating among the 'self' which resembles the normal operation of the monitored system and the 'non-self indicating abnormal data. For example, Ramdane et al. [133] developed a NS approach called Hybrid NSA for IDS Adaptation to build an effective NADS which was adapted automatically to be able to recognise low-footprint attacks.

5.6.2. Parametric approaches

Parametric approaches assume that network data follow a certain distribution, for instance, that a Gaussian distribution estimates the parameters of the given data [2, 3, 18, 41, 53]. However as, in real networking, the underlying distribution of network traffic data is not known, it is important to specify which probability distribution can fit the data with a relatively low error rate.

It is observed that network data do not belong to a Gaussian distribution [75] using The Kolmogorov-Smirnov (K-S) test method [75], it is better to apply non-Gaussian distributions, such as a Gaussian Mixture Model (GMM), Beta Mixture Model (BMM) or Dirichlet Mixture model (DMM), to network data. The Probability Density Functions (PDFs) of these distributions have to be

modelled from the ingress network data from which their parameters should be dynamically adjusted, instead of there being a static setting, to build a flexible model which distinguishes anomalies from normal observations [3, 128].

The methodologies of the most commonly used parametric methods are discussed in the following.

- **Particle filter**

  A particle filter is an inference mechanism which measures the unknown state from a set of observations with respect to a time, with the posterior distribution established by a set of weighted particles [134, 135]. For example, Xu et al. [136] proposed a Continuous Time BN (CTBN) model for detecting attacks that penetrated both host and network activities.

- **Bayesian network (BN)**

  A BN is a graphical probability distribution for making decisions regarding uncertain data [52]. For instance, Altwaijry [137] developed a naive BN NADS using the PCA which computed the highest ranked features within the PCA and used the selected features and their components as weights to improve the traditional naive Bayesian technique. The experimental results reflected that it could effectively decrease the data dimensions and improve detection accuracy. Han et al. [138] designed a NADS using a combination of a naive BN classifier, Linear Discriminant Analysis (LDA) and chi-square feature selection.

- **Finite mixture models**

  As a finite mixture model can be defined as a convex combination of two or more PDFs, the joint properties of which can approximate any arbitrary distribution, it is a powerful and flexible probabilistic modelling tool for univariate and multivariate data [2, 3, 5, 41, 139]. Network data are typically considered multivariate as they have d dimensions for differentiating between attack and normal instances [2, 3, 75]. The GMM is the mixture model most often applied for NADSs. It estimates the PDF of the target class (i.e., normal class) given by a training set and is typically based on a set of kernels rather than the rules in the training phase [18, 41]. Mixture models require a large number of normal instances to correctly estimate their parameters and it is difficult to select a suitable threshold (δ), as in equation (12), which differentiates attack instances from the normal training class with a certain score.

$$\left\{ \begin{array}{lcl} \delta \geq score & \Longrightarrow & normal\ instance \\ otherwise & \Longrightarrow & anomalous\ instance \end{array} \right\} \tag{3}$$

This score can be defined using the unconditional probability distribution ($w(X) = p(x)$) and a typical approach for setting the threshold ($\delta = p(x)$) [140]. For example, Fan et al. [141] developed an unsupervised statistical technique for identifying network intrusions in which legitimate and anomalous patterns were learned through finite generalised Dirichlet mixture models based on Bayesian inference, with the parameters of the mixture model and feature saliency simultaneously estimated.

Greggio [142] designed a NADS based on the unsupervised fitting of network data using a GMM which selected the number of mixture components

Table 6: Comparison of decision engine mechanisms

DE techniques	Related studies	Advantages	Disadvantages
Classification	[92], [144], [88], [96], [104], [145], [52]	- produces high detection rate and low false positive rate if the network data is correctly labelled	- depends on the assumption that each classifier has to be constructed separately
			- takes more computational resources
Clustering	[146], [103], [101], [147], [100], [102], [13]	- groups data with no dependency on the class label	- depends on the efficacy of establishing a legitimate profile
		- decreases processing times	- needs a higher time while updating the established profile
Knowledge	[148], [149],[150], [118][88], [120]	- identifies on known intrusive activities	- consumes too much time during the training and testing phases
		- provides a high detection rate for existing attacks	- applies static rules for recognising suspicious events
Combination	[52],[151], [127], [119], [124],[126]	- attains high accuracy and detection rates	- demands a huge effort to incorporate more than one technique
		- needs only a set of controlling parameters to be adapted.	- consumes a long processing time than other mechanisms
Statistics	[22] [152], [131], [130],[133], [2],[136], [3]	- achieve higher accuracy and detection rates if a threshold of identifying attacks correctly adjusted from network data, as provided in this thesis	- need precise analysis to select the correct threshold
		- do not take computational resources like other mechanisms	- demand new functions to identify attack types, such DoS and DDoS

and fit the parameter for each component in a real environment. The highest covariance matrix identified legitimate network activities, with the smaller components treated as anomalies. Christian et al. [143] proposed a NADS based on combining parametric and non-parametric density modelling mechanisms in two steps. Firstly, malicious samples were recognised using the GMM and then clustered in a non-parametric measure in the second step. While a cluster stretched to an adequate size, a procedure was identified, transformed into a parametric measure and added to the established GMM. These techniques were evaluated using the KDD99 dataset and their results reflected a high detection accuracy and low FPR. However, they would require the use of Bayesian inference to be adjusted for their efficient application in real networking.

A brief comparison between advantages and disadvantage of the existing DE techniques is demonstrated in Table 6.

Table 7: Confusion matrix for binary classification problems

		Actual	
		Negative	Positive
Predicted	Negative	TN	FP
	Positive	FN	TP

6. Evaluation metrics for IDS

The evaluation criteria of an IDS depends on estimating a confusion matrix as a classification problem demonstrated in Table 7 [23]. The purpose of the confusion matrix is to compare actual and predicted labels. It is acknowledged that an intrusion detection problem contains two classes: normal and attack, which is defined by a 2-by-2 confusion matrix for an evaluation.

The terms TP (true positive) and TN (true negative) denote correctly predicted conditions and FP (false positive) and FN (false negative) misclassified ones. TPs and TNs refer to correctly classified attack and normal records, respectively and, conversely, FPs and FNs refer to misclassified normal and attack records, respectively [2, 14, 23]. These four terms are used to generate the following IDS evaluation measures.

- **Accuracy** is a metric that estimates the overall percentages of detection and false alarms an IDS model produces, which reflects the overall success rate of any IDS, and is computed as

$$Accuracy = (TN + TP)/(TP + FP + TN + FN) \tag{4}$$

- **The Detection Rate (DR)**, also called the true positive rate (TPR) or sensitivity, is the proportion of correctly classified malicious instances of the total number of malicious vectors and is computed as

$$DR = TP/(FN + TP) \tag{5}$$

- **The True Negative Rate (TNR)**, also called the specificity, is the percentage of correctly classified normal instances of the total number of normal vectors and is computed as

$$TNR = TN/(TN + FP) \tag{6}$$

- **The False Positive Rate (FPR)** is the percentage of normal vectors of the total number of normal vectors misclassified as attacks and is computed as

$$FPR = FP/(FP + TN) \tag{7}$$

- **The False Negative Rate (FNR)** is the percentage of misclassified attack vectors of the total number of attack instances, given as

$$FNR = FN/(FN + TP) \tag{8}$$

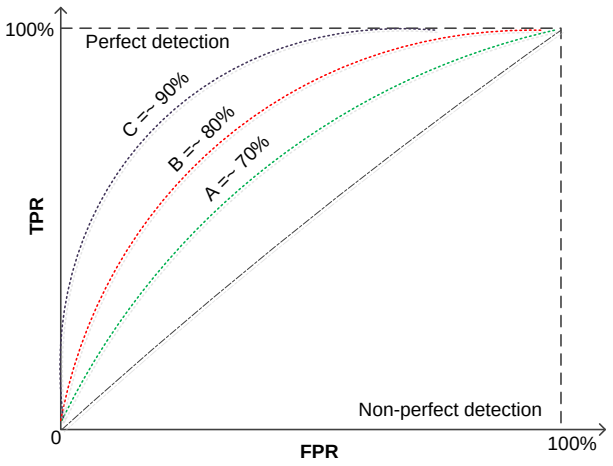

Figure 13: ROC curves - A, B and C show levels of detection

IDS approaches are evaluated using the TPR-FNR or sensitivity-specificity measure to estimate to what extent they are accurate in detecting malicious activities [23]. A perfect IDS approach could have a 100% DR while a 0% FPR reflects that all attack instances are detected without any misclassification. However, this is very difficult and demonstrates the optimal performance to be achieved in a real environment. Sensitivity gets more priority when the system is protected at costs of obtaining high false positive and negative rates while specificity gains high priority when accuracy is too low [23]. The accuracy measure is not a useful metric for IDSs because intrusion detection data is usually unbalanced, where there are much more legitimate data instances than malicious ones.

Another measure commonly used is the Receiver Operating Characteristics (ROC) curve. It was created from the signal processing theory and then extended to other domains, such as data mining and machine learning as well as artificial intelligence. In an intrusion detection methodology, it represents the relationship between the TPR and FPR of a DE approach [12, 23], as shown in Figure 13. The curve C is better than the curves B and A, as the ROC value is closer to 100%, which is the perfect detection rate.

the F-measure criterion is a preferable measure of evaluating IDS approaches. It is a harmonious mean of precision and recall [153], that is, a statistical function for estimating the accuracy of a system by computing its precision and recall given as

$$F - measure = 2 * (Precision * Recall)/(Precision + Recall) \qquad (9)$$

where precision is the fraction of the predicted positive values which are actually positive and recall the actual number of positives correctly detected, as given in equations (7) and (8), respectively.

$$Precision = TP/(TP + FP) \tag{10}$$

$$Recall = TP/(TP + FN) \tag{11}$$

Similar to the TRP-FPR measure, when the precision and recall of an IDS approach achieve 100%, as the F-measure is the maximum, a 0% FAR and 100% DR are produced [153, 154]. There are also other measures that could be used for estimating the efficiency and reliability of IDSs, as listed below ([23, 155]).

- **Performance** – is the capability of a system to handle network traffic that deals with a high speed and low packet loss while running in real time. As, in a real network environment, the packets are different sizes, the efficacy of an IDS relies on its capability to process a packet of any size. Moreover, CPU and memory usage could also be considered criteria for assessing an IDS performance [12, 23]. The performance of any IDS depends on its configuration in a network and the capacity of the network it monitors.

- **Completeness** - is the capability to detect all the vulnerabilities and attacks that attempt to breach a network [12]. This measure is more difficult to appraise than the others as it is impossible to have knowledge about malicious activities which could penetrate a user's privileges.

- **Timeliness** - indicates the capability of an IDS to perform its inspection as quickly as possible to enable the security administrator or response engine to take action before a great deal of loss occurs [155, 156]. There is a continual delay between the detection of an attack and the response of the system which it is preferable to reduce as much as possible to prevent attack threats.

- **Profile update** – when new vulnerabilities or abuses are identified, black-lists or profiles have to be updated for new detection [156]. However, this task is a big challenge in current high-speed network traffic for distinguishing between normal and attack events [2, 3].

- **Stability** - an IDS should operate consistently in different network infrastructures and steadily log identical events to allow its triggers to be easily configured [23].

- **Interoperability** - an effective IDS is assumed to be capable of associating information from numerous sources, such as system and firewall logs, HIDSs, NIDSs and any other available source of information [157].

7. Feature selection and decision engine evaluations

In order to explain how the feature selection and decision engine approaches can be applied to NIDSs using some existing datasets, this section discusses the effective role of feature selection techniques in improving the performances of DE approaches. We applied the ARM, PCA and ICA techniques, which have been widely used in the last few years, on the KDD99/NSL-KDD and UNSW-NB15 datasets. The ARM technique was used as an example of a wrapper

Table 8: Features selected from both datasets

Selected features	Description
NSL-KDD dataset	
srv_count	Number of connections to the same service as the current connection in the past two seconds
dst_host_srv_count	Number of connections to the same service in the past 100 connections
count	Number of connections to the same host as the current connection in the past two seconds
dst_host_same_srv rate	Number of connections to different service as the current connection in the past two seconds
dst_host_count	Number of connections to the same host in the past 100 connections
hot	Hot indicators, e.g., access to system directories, creation, and execution of programs
srv_diff_host_rate	Percentage of same service connections to different hosts.
rerror_rate	Percentage of same host connections that have “REJ" errors
UNSW-NB15 dataset	
ct_dst_sport_ltm	Number of connections containing the same destination address and source port in 100 connections
tcprtt	Round-trip time of TCP connection setup computed by the sum of 'synack' and 'ackdat'
dwin	Value of destination TCP window advertisement.
ct_src_dport_ltm	Number of connections containing the same source address and destination port in 100 connections
ct_dst_src_ltm	Number of connections containing the same source and destination address in 100 connections
ct_dst_ltm	Number of connections containing the same destination address in 100 connections
smean	Mean of flow packet sizes transmitted from source
service	Service types, e.g., HTTP, FTP, SMTP, SSH, DNS and IRC

FS method that depends on labels, while the PCA and ICA techniques were utilised as filter FS methods without labels. Moreover, the three techniques can effectively deal with the potential characteristics of network data such as non-linear and non-normal distributions [2, 3, 41, 78].

The techniques were developed using the 'R programming language' on Linux Ubuntu 14.04 with 16 GB RAM and an i7 CPU processor. To conduct the experiments on each dataset, we select random samples from them with different sample sizes of between 50,000 and 250,000. For each sample size used to establish the normal profile (i.e., the training phase), each normal sample is almost 65-75% of the total size while the others are used in the testing phase which establishes the principle of NADS on which we focus in this paper. The performances of the DE techniques are evaluated using 10-fold cross–validations of the sample sizes to determine their effects on all samples included in the learning and validation processes.

The most important features are selected from the rules of the ARM technique which have higher levels of importance, and from the components of the PCA and ICA with higher variances. The eight features for each dataset listed in Table 8 are selected to reduce the processing time while applying DE as,

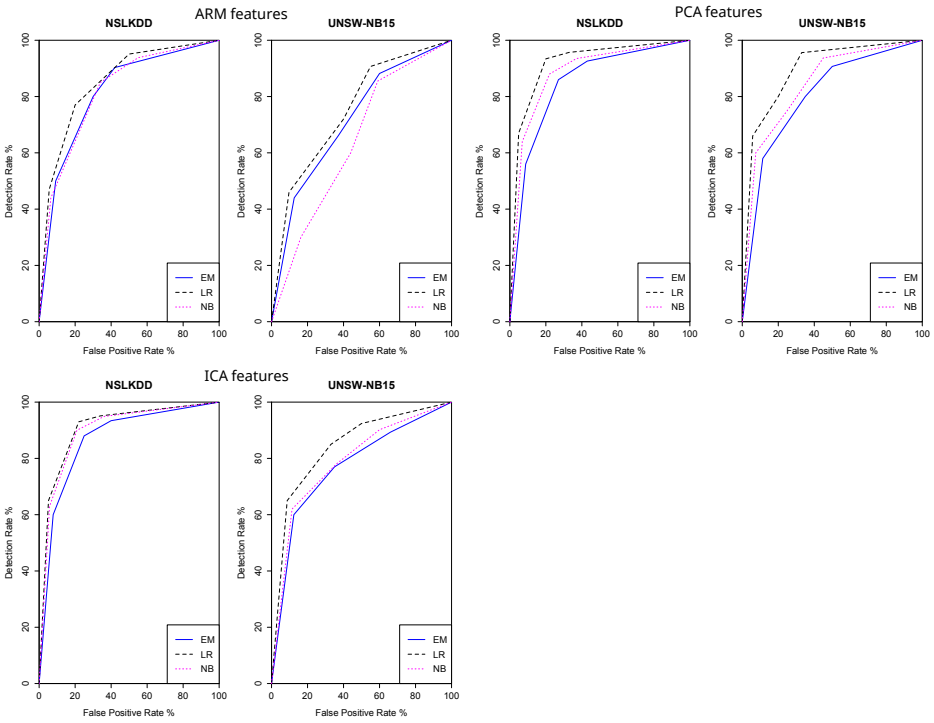

Figure 14: ROC curves of three ML algorithms using ARM, PCA and ICA FS techniques

for less than this number, DE evaluations provide lower accuracies and higher FARs [3, 77, 78].

To assess performances using the features selected from the datasets, three ML algorithms, namely, EM clustering, Logistic Regression (LR) and Naive Bayes (NB), are applied. The EM clustering technique was used as an example of unsupervised learning that can identify attacks without using labels in the training phase, while the LR and NB techniques were utilised as examples of statistical and supervised learning approaches that demand labels to classify attacks and their types. The evaluation criteria are estimated in terms of the accuracy, and FAR and ROC curves to assess the effects of these features and how they could improve performances at a lower computational cost, with the results obtained provided in Table 9.

There are two reasons for the ML algorithms performing better on the KDD99/NSL- KDD than UNSW-NB15 dataset. Firstly, the latter has many values of normal and suspicious instances that are almost the same while the former does not. Secondly, the data distributions of the NSL-KDD dataset's training and testing sets are different due to the insertion of new attacks into the testing set which clearly distinguish between its normal and abnormal instances while executing ML algorithms. However, these distributions are approximately the same in the UNSW-NB15 dataset because its normal and abnormal instances were created from the same network. To compare the results obtained from the three FS methods, we observe that the last two often provide better evaluation results than the ARM using ML algorithms, as shown in Figures 14.

Table 9: Performance evaluation using both datasets

Techniques	**ARM**				**PCA**			
	NSL-KDD		UNSW-NB15		NSL-KDD		UNSW-NB15	
	Accuracy	FAR	Accuracy	FAR	Accuracy	FAR	Accuracy	FAR
	(%)	(%)	(%)	(%)	(%)	(%)	(%)	(%)
EM	90.3	9.2	88.2	12.6	93.4	7.8	89.3	12.4
LR	95.1	5.6	90.7	9.7	95.1	5.2	92.4	8.7
NB	93.8	6.5	85.5	16.3	94.9	5.8	90.2	11.4

Techniques	**ICA**			
	NSL-KDD		UNSW-NB15	
	Accuracy	FAR	Accuracy	FAR
	(%)	(%)	(%)	(%)
EM	92.6	8.8	90.7	11.8
LR	95.7	4.9	95.6	5.8
NB	93.5	6.9	93.7	7.5

Table 10: Comparison of feature selection and DE approaches on various datasets

DE approach	**Technique**	**FS method**	**Accuracy (%)**	**FAR (%)**	**Dataset**
Classification	KNN [158]	PCA	80.6	11.4	KDD99
	Naive Bayes [159]	Information gain	82.5	17.3	Kyoto
	Fuzzy technique [160]	Fuzzy extractor	92.8	8.1	NGIDS-DS
Clustering	Optimum-path clustering [161]	Particle Swarm	96.1	3.4	ISCX
	SOM clustering [161]	Particle Swarm	99.8	0.2	NSL-KDD
	A Semantic Approach [162]	Association rules	91.7	8.7	ADFA
	Genetic algorithm [163]	-	94.3	5.6	NSL-KDD
Combination	Ensemble classifier [159]	Information gain	90.5	0.2	Kyoto
	Ramp-KSVCR [164]	Correlation coefficient	93.5	2.7	UNSW-NB15
	Ramp-KSVCR [164]	Correlation coefficient	98.8	0.9	NSL-KDD
Statistics	GAA [3]	PCA	92.8	5.1	UNSW-NB15
	Bayesian network [165]	Chow–Liu algorithm	89.3	10.7	ISCX

This is because the ARM technique deals directly with the values of features while the others transform the feature space into another space based on the highest variances between features which can greatly help DE techniques find differences between normal and suspicious instances. However, the ARM method can provide promising results when selecting relevant observations. Regarding the PCA and ICA techniques, there are only small differences in the evaluation performances of the ML algorithms as their internal methodologies appear to be similarly based on variances. Consequently, we suggest using the PCA in the feature reduction model due to its simplicity of execution and better performances using ML algorithms [2, 3, 140].

In order to provide fair comparisons between the datasets in terms of the FS and DE approaches discussed above, Table 10 presents some recently published techniques. It is observed that FS methods can significantly improve the performance of a NADS by excluding irrelevant attributes from datasets. NADSs using different DE approaches have their own merits and demerits, as shown in Table 6. As statistical and ML techniques constantly try to enhance the process of detecting abnormal activities from network and host systems, their complexity becomes one of the essential criteria that should be considered in the design of a lightweight and reliable NADS. For learning and validating ML mechanisms on new datasets, combination and statistical techniques can effectively detect existing and zero-day attacks while knowledge, classification and clustering can efficiently detect known ones.

The DE approaches used to identify recent network threats are explained in Section 5. Classification, statistical and clustering algorithms can generally discover DoS, DDoS and botnet attacks because they can learn from the massive amounts of data hackers send to victims' systems. They can also discriminate between DDoS and Flash crowded based on their different characteristics [166]. Knowledge and classification techniques can recognise brute force and shellshock malicious events as they can detect attempts to penetrate users' credentials and/or remotely exploit systems [22]. Clustering algorithms can detect browser-based attacks because they can group legitimate rules generated from websites and identify outliers as attacks [42, 75]. Combination and classification mechanisms can effectively identify SSL anomalous behaviours because they can deal properly with features extracted from TLS/SSL protocols and achieve promising detection rates [22]. Finally, statistical and classification techniques can recognise backdoor attacks by effectively identifying abnormal patterns of the IRC protocol.

8. Challenges and future directions

Although a MDS cannot recognise future attacks or variants of existing attack types, it is still a common defence solution used in commercial products. On the contrary, a NADS can detect serious threats but has often been faced with potential challenges for its effective design. These challenges, which can be explored from an anomaly-based methodology [6, 13, 23, 75, 140], are as follows.

- Constructing a comprehensive profile that involves all possible legitimate behaviours is very complex as the boundary between normal and abnormal behaviours is usually not accurate because the network features selected cannot reflect any variations between normal and abnormal patterns using

detection methods. FPR and FNR errors occur when a normal behaviour falls in an attack region and a malicious one in a normal region, respectively.

- When designing the architecture of an adaptive and scalable NADS, autonomous NADS techniques that can handle the large sizes and high speeds of current network systems should be used because they are automatically capable of adapting their threshold, that is, the baseline between legitimate and attack events.

- Real-time detection is also very challenging for several reasons. Firstly, the features created for network traffic may contain a set of noisy or irrelevant ones. Secondly, the lightweight of detection methods need to be carefully adopted, with respect to the above problems. These reasons increase the processing time and false alarm rate if not properly addressed. Therefore, feature reduction and lightweight DE approaches should be developed. The feature reduction will assist in reducing irrelevant attributes, and DE approaches will improve the detection accuracy if they can discriminate between the low variations of normal and abnormal patterns.

- The availability of a good public IDS dataset is usually a major concern for learning and validating NADS models. This is because such datasets should have a broad range of contemporary normal and malicious behaviours as well as being correctly labelled, which is difficult. Most of the existing IDS datasets often suffer from inaccurate labelling, poor attack diversity, and incomplete network information capture without including both headers and payloads. Creating new IDS datasets demand designing realistic environments that include different normal and attack scenarios. Moreover, the ground truth that includes attack events should be generated to trust the dataset's credibility in testing new IDSs

- Since new types of ransomware has recently increased, organisations face a high risk to protect their assets. Ransomware is malware that harms computer and network systems by encrypting computer resources and blocking access till a ransom is paid. The first execution involves malicious scripts that has to communicate with a C&C server to receive the encryption key. Designing flow features is essential as the payload of packets are encrypted. Moreover, developing efficient feature selection and flow aggregation methods capable of reducing large sizes of network traffic could assist in discovering ransomware attacks.

- Designing effective ADSs that can efficiently identify future cyber adversaries from IoT, Cloud/Fog computing paradigms, industrial control systems, or Software Defined Networks. New ADSs should be able to monitor high-speed networks that exchange high data rates in real-time. Moreover, such systems should be scalable and self-adaptive for protecting different nodes of wide area networks. In IoT networks, there is a large amount of network traffic and telemetry data of IoT sensors as well as Cloud/Fog services that should be examined [41, 43, 46]. Moreover, this requires building collaborative NADSs to analyse different network nodes and aggregate their data for recognising suspicious events.

- Identifying cyber espionage attacks through data exfiltration is one of the major challenges in IoT networks [25]. The attacker transmits targeted data to exploit IoT sensors and network platforms. These malicious activity can not be detected using traditional NADSs, as it is essential to design profiles of normal events for telemetry data of IoT sensors and network traffics. The utilisation of mixture models and deep learning algorithms could improve the NADS's performance. But, this will create issues related to handling the collected big data, interoperability and scalability in order to design an effective and real-time NADSs.

9. Concluding remarks

This study discussed the background and literature related to IDSs, specifically a NADS with different applications of backbone, IoT, data centers, Cloud and Fog Computing paradigms. Due to rapid advances in technologies, computer network systems need a solid layer of defence against vulnerabilities and severe threats. Although an IDS is a significant cyber security application which integrates a defence layer to achieve secure networking, it still faces challenges for being built in an online and adaptable manner. Anomaly detection methodologies which can efficiently identify known and zero-day attacks are investigated. It has been a very challenging issue to apply a NADS instead of a MDS methodology in the computer industry which could be overcome by framing its architecture with a data source, pre-processing method and DE mechanism.

A NADS is usually evaluated on a data source/dataset which involves a wide variety of contemporary normal and attack patterns that reflect the performances of DE approaches. The network dataset used consists of a set of features and observations that may include irrelevant ones that could negatively affect the performances and accuracy of DE approaches. Consequently, data pre-processing methods for creating, generating, reducing, converting and normalising features are discussed to pass filtered information to a DE approach which distinguishes between anomalous and legitimate observations and has been applied based on classification, clustering, knowledge, combination and statistics discussed to demonstrate their merits and demerits in terms of building an effective NADS.

References

[1] A. Shameli-Sendi, M. Cheriet, A. Hamou-Lhadj, Taxonomy of intrusion risk assessment and response system, Computers & Security 45 (2014) 1–16.

[2] N. Moustafa, G. Creech, J. Slay, Big data analytics for intrusion detection system: statistical decision-making using finite dirichlet mixture models, in: Data Analytics and Decision Support for Cybersecurity, Springer, 2017, pp. 127–156.

[3] N. Moustafa, J. Slay, G. Creech, Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks, IEEE Transactions on Big Data (2017) 1–14.

[4] N. Moustaf, J. Slay, Creating novel features to anomaly network detection using darpa-2009 data set, in: Proceedings of the 14th European Conference on Cyber Warfare and Security, 2015, p. 204.

[5] N. Moustafa, G. Creech, J. Slay, Anomaly detection system using beta mixture models and outlier detection, in: Progress in Computing, Analytics and Networking, Springer, 2018, pp. 125–135.

[6] S. Pontarelli, G. Bianchi, S. Teofili, Traffic-aware design of a high-speed fpga network intrusion detection system, IEEE Transactions on Computers 62 (11) (2013) 2322–2334.

[7] L. Wang, R. Jones, Big data analytics for network intrusion detection: A survey, International Journal of Networks and Communications 7 (1) (2017) 24–31.

[8] Z. Inayat, A. Gani, N. B. Anuar, M. K. Khan, S. Anwar, Intrusion response systems: Foundations, design, and challenges, Journal of Network and Computer Applications 62 (2016) 53–74.

[9] S. Anwar, J. Mohamad Zain, M. F. Zolkipli, Z. Inayat, S. Khan, B. Anthony, V. Chang, From intrusion detection to an intrusion response system: Fundamentals, requirements, and future directions, Algorithms 10 (2) (2017) 39.

[10] B. B. Zarpelao, R. S. Miani, C. T. Kawakani, S. C. de Alvarenga, A survey of intrusion detection in internet of things, Journal of Network and Computer Applications 84 (2017) 25–37.

[11] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, E. Vázquez, Anomaly-based network intrusion detection: Techniques, systems and challenges, computers & security 28 (1) (2009) 18–28.

[12] I. Corona, G. Giacinto, F. Roli, Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues, Information Sciences 239 (2013) 201–225.

[13] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM computing surveys (CSUR) 41 (3) (2009) 15.

[14] M. Ahmed, A. N. Mahmood, J. Hu, A survey of network anomaly detection techniques, Journal of Network and Computer Applications 60 (2016) 19–31.

[15] A. A. Aburomman, M. B. I. Reaz, A survey of intrusion detection systems based on ensemble and hybrid classifiers, Computers & Security 65 (2017) 135–152.

[16] J. Peng, K.-K. R. Choo, H. Ashman, User profiling in intrusion detection: A review, Journal of Network and Computer Applications 72 (2016) 14–27.

[17] N. Moustafa, B. Turnbull, K. R. Choo, An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things, IEEE Internet of Things Journal (2018) 1–1doi:10.1109/JIOT.2018.2871719.

[18] N. Moustafa, G. Creech, E. Sitnikova, M. Keshk, Collaborative anomaly detection framework for handling big data of cloud computing, in: Military Communications and Information Systems Conference (MilCIS), 2017, IEEE, 2017, pp. 1–6.

[19] S. Sharma, A. Kaul, A survey on intrusion detection systems and honeypot based proactive security mechanisms in vanets and vanet cloud, Vehicular Communications (2018) 138–164.

[20] P. A. A. Resende, A. C. Drummond, A survey of random forest based methods for intrusion detection systems, ACM Computing Surveys (CSUR) 51 (3) (2018) 48.

[21] T. Sager, Killing advanced threats in their tracks: An intelligent approach to attack prevention, SANS Institute, Tech. Rep (2014) 1–17.

[22] G. Creech, Developing a high-accuracy cross platform host-based intrusion detection system capable of reliably detecting zero-day attacks, Ph.D. thesis, University of New South Wales, Australia (2014).

[23] M. H. Bhuyan, D. K. Bhattacharyya, J. K. Kalita, Network anomaly detection: methods, systems and tools, IEEE Communications Surveys & Tutorials 16 (1) (2014) 303–336.

[24] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, K.-Y. Tung, Intrusion detection system: A comprehensive review, Journal of Network and Computer Applications 36 (1) (2013) 16–24.

[25] The acsc threat report.
URL `https://www.acsc.gov.au/publications/`

[26] The macafee threat report.
URL `http://www.mcafee.com/us/resources/`

[27] P. Gasti, G. Tsudik, E. Uzun, L. Zhang, Dos and ddos in named data networking, in: 2013 22nd International Conference on Computer Communication and Networks (ICCCN), 2013, pp. 1–7. doi:10.1109/ICCCN.2013.6614127.

[28] S. Honda, Y. Unno, K. Maruhashi, M. Takenaka, S. Torii, Topase: Detection of brute force attacks used disciplined ips from ids log, in: Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on, IEEE, 2015, pp. 1361–1364.

[29] G. F. He, T. Zhang, Y. Y. Ma, J. X. Fei, Protecting users privacy from browser-based attacks, in: Applied Mechanics and Materials, Vol. 631, Trans Tech Publ, 2014, pp. 941–945.

[30] Y. Ji, X. Zhang, T. Wang, Backdoor attacks against learning systems, in: Communications and Network Security (CNS), 2017 IEEE Conference on, IEEE, 2017, pp. 1–9.

[31] A. L. Buczak, E. Guven, A survey of data mining and machine learning methods for cyber security intrusion detection, IEEE Communications Surveys & Tutorials 18 (2) (2016) 1153–1176.

[32] E. Hodo, X. Bellekens, A. Hamilton, C. Tachtatzis, R. Atkinson, Shallow and deep networks intrusion detection system: A taxonomy and survey, arXiv preprint arXiv:1701.02145.

[33] D. Moon, S. B. Pan, I. Kim, Host-based intrusion detection system for secure human-centric computing, The Journal of Supercomputing 72 (7) (2016) 2520–2536.

[34] M. Keshk, N. Moustafa, E. Sitnikova, G. Creech, Privacy preservation intrusion detection technique for scada systems, in: Military Communications and Information Systems Conference (MilCIS), 2017, IEEE, 2017, pp. 1–6.

[35] R. Bar-Yanai, M. Langberg, D. Peleg, L. Roditty, Realtime classification for encrypted traffic, in: International Symposium on Experimental Algorithms, Springer, 2010, pp. 373–385.

[36] H. A. Kholidy, F. Baiardi, Cids: A framework for intrusion detection in cloud systems, in: Information Technology: New Generations (ITNG), 2012 Ninth International Conference on, IEEE, 2012, pp. 379–385.

[37] M. Kumar, M. Hanumanthappa, T. Suresh Kumar, Encrypted traffic and ipsec challenges for intrusion detection system, in: Proceedings of International Conference on Advances in Computing, Springer, 2013, pp. 721–727.

[38] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, E. Vázquez, Anomaly-based network intrusion detection: Techniques, systems and challenges, computers & security 28 (1) (2009) 18–28.

[39] N. Moustafa, G. Misra, J. Slay, Generalized outlier gaussian mixture technique based on automated association features for simulating and detecting web application attacks, IEEE Transactions on Sustainable Computingdoi:10.1109/TSUSC.2018.2808430.

[40] S. Kaur, M. Singh, Automatic attack signature generation systems: A review, IEEE Security & Privacy 11 (6) (2013) 54–61.

[41] N. Moustafa, E. Adi, B. Turnbull, J. Hu, A new threat intelligence scheme for safeguarding industry 4.0 systems, IEEE Access 6 (2018) 32910–32924.

[42] A. Milenkoski, M. Vieira, S. Kounev, A. Avritzer, B. D. Payne, Evaluating computer intrusion detection systems: A survey of common practices, ACM Computing Surveys (CSUR) 48 (1) (2015) 12.

[43] H. H. Pajouh, R. Javidan, R. Khayami, D. Ali, K.-K. R. Choo, A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in iot backbone networks, IEEE Transactions on Emerging Topics in Computing (2016) 1–11.

[44] D. McGrew, T. Rigoudy, Intrusion detection to prevent impersonation attacks in computer networks, uS Patent App. 15/616,514 (Sep. 21 2017).

[45] I. Figlin, A. Zavalkovsky, L. Arzi, E. Hudis, J. R. LeMond, R. E. Fitzgerald, K. E. Ahmed, J. S. Williams, E. W. Hardy, Network intrusion detection with distributed correlation, uS Patent 9,560,068 (Jan. 31 2017).

[46] E. Benkhelifa, T. Welsh, W. Hamouda, A critical review of practices and challenges in intrusion detection systems for iot: Towards universal and resilient systems, IEEE Communications Surveys & Tutorials (2018) 1–15.

[47] J. F. Colom, D. Gil, H. Mora, B. Volckaert, A. M. Jimeno, Scheduling framework for distributed intrusion detection systems over heterogeneous network architectures, Journal of Network and Computer Applications 108 (2018) 76–86.

[48] R. Roman, J. Lopez, M. Mambo, Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges, Future Generation Computer Systems 78 (2018) 680–698.

[49] M. K. Marhas, A. Bhange, P. Ajankar, Anomaly detection in network traffic: A statistical approach, International Journal of IT, Engineering and Applied Sciences Research (IJIEASR) 1 (3) (2012) 16–20.

[50] The darpa98 and kddcup99 datasets.
URL `http://www.ll.mit.edu/ideval/data/1998data.html`

[51] The unsw-nb15 dataset.
URL `https://www.unsw.adfa.edu.au/australian-centre-for-cyber-security/cybersecurity/ADFA-NB15-Datasets/`

[52] S. Dua, X. Du, Data mining and machine learning in cybersecurity, 1st Edition, Vol. 1, CRC press, 2016.

[53] A. Vasudevan, E. Harshini, S. Selvakumar, Ssenet-2011: a network intrusion detection system dataset and its comparison with kdd cup 99 dataset, in: Internet (AH-ICI), 2011 Second Asian Himalayas International Conference on, IEEE, 2011, pp. 1–5.

[54] A. Patcha, J.-M. Park, An overview of anomaly detection techniques: Existing solutions and latest technological trends, Computer networks 51 (12) (2007) 3448–3470.

[55] R. Zuech, T. M. Khoshgoftaar, R. Wald, Intrusion detection and big heterogeneous data: a survey, Journal of Big Data 2 (1) (2015) 1.

[56] The hadoop technologies.
URL `http://hadoop.apache.org/`

[57] The mysql cluster cge technology.
URL https://www.mysql.com/products/cluster/

[58] N. Moustafa, G. Creech, J. Slay, Flow aggregator module for analysing network traffic, in: International Conference on Computing Analytics and Networking(ICCAN 2017), Springer, 2017.

[59] The nslkdd dataset.
URL https://web.archive.org/web/20150205070216/http://nsl.cs.unb.ca/NSL-KDD/

[60] The caida datasets.
URL https://www.caida.org/data/

[61] The defcon dataset.
URL http://www.netresec.com/?page=PcapFiles

[62] The unibs dataset.
URL http://netweb.ing.unibs.it/ ntw/tools/traces/

[63] The lbnl dataset.
URL http://powerdata.lbl.gov/download.html

[64] M. H. Bhuyan, D. K. Bhattacharyya, J. K. Kalita, Towards generating real-life datasets for network intrusion detection., IJ Network Security 17 (6) (2015) 683–701.

[65] The darpa-2009 dataset. darpa scalable network monitoring (snm) program traffic. packet clearing house. 11/3/2009 to 11/12/2009.
URL https://www.predict.org/

[66] The cdx datasets.
URL https://www.usma.edu/crc/SitePages/DataSets.aspx

[67] The ctu-13 dataset.
URL https://www.usma.edu/crc/SitePages/DataSets.aspx

[68] A. Shiravi, H. Shiravi, M. Tavallaee, A. A. Ghorbani, Toward developing a systematic approach to generate benchmark datasets for intrusion detection, computers & security 31 (3) (2012) 357–374.

[69] The iscx dataset.
URL http://www.unb.ca/research/iscx/dataset/iscx-IDS-dataset.html

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

[71] The adfa intrusion detection datasets.
URL https://www.unsw.adfa.edu.au/australian-centre-for-cyber-security/cybersecurity/ADFA-IDS-Datasets/

[72] N. Moustafa, J. Slay, Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set), in: Military Communications and Information Systems Conference (MilCIS), 2015, IEEE, 2015, pp. 1–6.

[73] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, Toward generating a new intrusion detection dataset and intrusion traffic characterization., in: ICISSP, 2018, pp. 108–116.

[74] W. Haider, J. Hu, J. Slay, B. Turnbull, Y. Xie, Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling, Journal of Network and Computer Applications 87 (2017) 185–192.

[75] N. Moustafa, J. Slay, The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set, Information Security Journal: A Global Perspective 25 (1-3) (2016) 18–31.

[76] T. Baba, S. Matsuda, Tracing network attacks to their sources, IEEE Internet Computing 6 (2) (2002) 20–26.

[77] N. Moustafa, J. Slay, A hybrid feature selection for network intrusion detection systems: Central points, in: Security Research Institute, Edith Cowan University, Australia, Vol. The Proceedings of the 16th Australian Information Warfare Conference, 2015, pp. 5–13.

[78] N. Moustafa, J. Slay, The significant features of the unsw-nb15 and the kdd99 data sets for network intrusion detection systems, in: Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS), 2015 4th International Workshop on, IEEE, 2015, pp. 25–31.

[79] H. Liu, H. Motoda, Feature selection for knowledge discovery and data mining, Vol. 454, Springer Science & Business Media, 2012.

[80] Y. Chen, Y. Li, X.-Q. Cheng, L. Guo, Survey and taxonomy of feature selection algorithms in intrusion detection system, in: International Conference on Information Security and Cryptology, Springer, 2006, pp. 153–167.

[81] Y. Zhao, S. S. Bhowmick, Association rule mining with r, A Survey Nanyang Technological University, Singapore (2015) 1–13.

[82] P. Xanthopoulos, P. M. Pardalos, T. B. Trafalis, Principal component analysis, in: Robust data mining, Springer, 2013, pp. 21–26.

[83] F. Palmieri, U. Fiore, A. Castiglione, A distributed approach to network anomaly detection based on independent component analysis, Concurrency and Computation: Practice and Experience 26 (5) (2014) 1113–1129.

[84] W. Lee, S. J. Stolfo, et al., Data mining approaches for intrusion detection., in: Usenix security, 1998.

[85] K. Nalavade, B. Meshram, Mining association rules to evade network intrusion in network audit data, International Journal of Advanced Computer Research 4 (2) (2014) 560.

[86] J. Luo, S. M. Bridges, Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection, International Journal of Intelligent Systems 15 (8) (2000) 687–703.

[87] Z. Yanyan, Y. Yuan, Study of database intrusion detection based on improved association rule algorithm, in: Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on, Vol. 4, IEEE, 2010, pp. 673–676.

[88] C. Wagner, J. François, T. Engel, et al., Machine learning approach for ip-flow record anomaly detection, in: International Conference on Research in Networking, Springer, 2011, pp. 28–39.

[89] L. Khan, M. Awad, B. Thuraisingham, A new intrusion detection system using support vector machines and hierarchical clustering, The VLDB Journal - The International Journal on Very Large Data Bases 16 (4) (2007) 507–521.

[90] M. Esmalifalak, H. Nguyen, R. Zheng, Z. Han, Stealth false data injection using independent component analysis in smart grid, in: Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on, IEEE, 2011, pp. 244–248.

[91] E. De la Hoz, E. De La Hoz, A. Ortiz, J. Ortega, B. Prieto, Pca filtering and probabilistic som for network intrusion detection, Neurocomputing 164 (2015) 71–81.

[92] I. Kang, M. K. Jeong, D. Kong, A differentiated one-class classification method with applications to intrusion detection, Expert Systems with Applications 39 (4) (2012) 3899–3905.

[93] N. Moustafa, J. Slay, A network forensic scheme using correntropy-variation for attack detection, in: IFIP International Conference on Digital Forensics, Springer, 2018, pp. 225–239.

[94] B. E. Boser, I. M. Guyon, V. N. Vapnik, A training algorithm for optimal margin classifiers, in: Proceedings of the fifth annual workshop on Computational learning theory, ACM, 1992, pp. 144–152.

[95] P. Poornachandran, S. Praveen, A. Ashok, M. R. Krishnan, K. Soman, Drive-by-download malware detection in hosts by analyzing system resource utilization using one class support vector machines, in: Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications, Springer, 2017, pp. 129–137.

[96] S.-J. Horng, M.-Y. Su, Y.-H. Chen, T.-W. Kao, R.-J. Chen, J.-L. Lai, C. D. Perkasa, A novel intrusion detection system based on hierarchical clustering and support vector machines, Expert systems with Applications 38 (1) (2011) 306–313.

[97] M. A. Ambusaidi, X. He, P. Nanda, Z. Tan, Building an intrusion detection system using a filter-based feature selection algorithm, IEEE transactions on computers 65 (10) (2016) 2986–2998.

[98] M. H. Bhuyan, D. K. Bhattacharyya, J. K. Kalita, Network traffic anomaly detection techniques and systems, in: Network Traffic Anomaly Detection and Prevention, Springer, 2017, pp. 115–169.

[99] M. Soltanolkotabi, E. J. Candes, et al., A geometric analysis of subspace clustering with outliers, The Annals of Statistics 40 (4) (2012) 2195–2238.

[100] H. Li, Research and implementation of an anomaly detection model based on clustering analysis, in: Intelligence Information Processing and Trusted Computing (IPTC), 2010 International Symposium on, IEEE, 2010, pp. 458–462.

[101] M. H. Bhuyan, D. Bhattacharyya, J. K. Kalita, An effective unsupervised network anomaly detection method, in: Proceedings of the International Conference on Advances in Computing, Communications and Informatics, ACM, 2012, pp. 533–539.

[102] M. H. Bhuyan, D. Bhattacharyya, J. K. Kalita, Nado: network anomaly detection using outlier approach, in: Proceedings of the 2011 International Conference on Communication, Computing & Security, ACM, 2011, pp. 531–536.

[103] G. Nadiammai, M. Hemalatha, An evaluation of clustering technique over intrusion detection system, in: Proceedings of the International Conference on Advances in Computing, Communications and Informatics, ACM, 2012, pp. 1054–1060.

[104] M. Saber, I. El Farissi, S. Chadli, M. Emharraf, M. G. Belkasmi, Performance analysis of an intrusion detection systems based of artificial neural network, in: Europe and MENA Cooperation Advances in Information and Communication Technologies, Springer, 2017, pp. 511–521.

[105] W. Huang, G. Song, H. Hong, K. Xie, Deep architecture for traffic flow prediction: deep belief networks with multitask learning, IEEE Transactions on Intelligent Transportation Systems 15 (5) (2014) 2191–2201.

[106] E. Hodo, X. Bellekens, A. Hamilton, C. Tachtatzis, R. Atkinson, Shallow and deep networks intrusion detection system: A taxonomy and survey, arXiv preprint arXiv:1701.02145.

[107] M. Z. Alom, V. Bontupalli, T. M. Taha, Intrusion detection using deep belief networks, in: Aerospace and Electronics Conference (NAECON), 2015 National, IEEE, 2015, pp. 339–344.

[108] B. Abolhasanzadeh, Nonlinear dimensionality reduction for intrusion detection using auto-encoder bottleneck features, in: Information and Knowledge Technology (IKT), 2015 7th Conference on, IEEE, 2015, pp. 1–5.

[109] C. Yin, Y. Zhu, J. Fei, X. He, A deep learning approach for intrusion detection using recurrent neural networks, IEEE Access 5 (2017) 21954–21961.

[110] S. A. Ludwig, Intrusion detection of multiple attack classes using a deep neural net ensemble, in: Computational Intelligence (SSCI), 2017 IEEE Symposium Series on, IEEE, 2017, pp. 1–7.

[111] N. Duffield, P. Haffner, B. Krishnamurthy, H. A. Ringberg, Systems and methods for rule-based anomaly detection on ip network flow, uS Patent 9,680,877 (Jun. 13 2017).

[112] K. Chadha, S. Jain, Hybrid genetic fuzzy rule based inference engine to detect intrusion in networks, in: Intelligent Distributed Computing, Springer, 2015, pp. 185–198.

[113] T. F. Lunt, R. Jagannathan, A prototype real-time intrusion-detection expert system, in: Security and Privacy, 1988. Proceedings., 1988 IEEE Symposium on, IEEE, 1988, pp. 59–66.

[114] The snort tool.
URL `https://www.snort.org/`

[115] H. Holm, Signature based intrusion detection for zero-day attacks:(not) a closed chapter?, in: System Sciences (HICSS), 2014 47th Hawaii International Conference on, IEEE, 2014, pp. 4895–4904.

[116] D. Midi, A. Rullo, A. Mudgerikar, E. Bertino, KalisŮa system for knowledge-driven adaptable intrusion detection for the internet of things, in: Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on, IEEE, 2017, pp. 656–666.

[117] B. Jasiul, M. Szpyrka, J. Śliwa, Malware behavior modeling with colored petri nets, in: IFIP International Conference on Computer Information Systems and Industrial Management, Springer, 2014, pp. 667–679.

[118] H. S. Vaccaro, G. E. Liepins, Detection of anomalous computer session activity, in: Security and Privacy, 1989. Proceedings., 1989 IEEE Symposium on, IEEE, 1989, pp. 280–289.

[119] P. Naldurg, K. Sen, P. Thati, A temporal logic based framework for intrusion detection, in: International Conference on Formal Techniques for Networked and Distributed Systems, Springer, 2004, pp. 359–376.

[120] S.-S. Hung, D. S.-M. Liu, A user-oriented ontology-based approach for network intrusion detection, Computer Standards & Interfaces 30 (1) (2008) 78–88.

[121] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42 (4) (2012) 463–484.

[122] D. B. Araya, K. Grolinger, H. F. ElYamany, M. A. Capretz, G. Bitsuamlak, An ensemble learning framework for anomaly detection in building energy consumption, Energy and Buildings 144 (2017) 191–206.

[123] V. Shah, A. K. Aggarwal, N. Chaubey, Performance improvement of intrusion detection with fusion of multiple sensors, Complex & Intelligent Systems (2016) 1–7.

[124] G. Folino, C. Pizzuti, G. Spezzano, An ensemble-based evolutionary framework for coping with distributed intrusion detection, Genetic Programming and Evolvable Machines 11 (2) (2010) 131–146.

[125] R. Perdisci, G. Gu, W. Lee, Using an ensemble of one-class svm classifiers to harden payload-based anomaly detection systems, in: Data Mining, 2006. ICDM'06. Sixth International Conference on, IEEE, 2006, pp. 488–498.

[126] H. H. Nguyen, N. Harbi, J. Darmont, An efficient local region and clustering-based ensemble system for intrusion detection, in: Proceedings of the 15th Symposium on International Database Engineering & Applications, ACM, 2011, pp. 185–191.

[127] A. A. Aburomman, M. B. I. Reaz, A novel svm-knn-pso ensemble method for intrusion detection system, Applied Soft Computing 38 (2016) 360–372.

[128] N. Shahid, I. H. Naqvi, S. B. Qaisar, Characteristics and classification of outlier detection techniques for wireless sensor networks in harsh environments: a survey, Artificial Intelligence Review 43 (2) (2015) 193–228.

[129] A. Soule, K. Salamatian, N. Taft, Combining filtering and statistical methods for anomaly detection, in: Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement, USENIX Association, 2005, pp. 31–31.

[130] X. Shen, S. Agrawal, Kernel density estimation for an anomaly based intrusion detection system., in: MLMTA, Citeseer, 2006, pp. 161–167.

[131] K. Caudle, C. Karlsson, L. D. Pyeatt, Using density estimation to detect computer intrusions, in: Proceedings of the 2015 ACM International Workshop on International Workshop on Security and Privacy Analytics, ACM, 2015, pp. 43–48.

[132] P. Mostardinha, B. F. Faria, A. Zúquete, F. V. de Abreu, A negative selection approach to intrusion detection, in: International Conference on Artificial Immune Systems, Springer, 2012, pp. 178–190.

[133] C. Ramdane, S. Chikhi, A new negative selection algorithm for adaptive network intrusion detection system, International Journal of Information Security and Privacy (IJISP) 8 (4) (2014) 1–25.

[134] C.-C. Lin, M.-S. Wang, Particle Filter for Depth Evaluation of Networking Intrusion Detection Using Coloured Petri Nets, INTECH Open Access Publisher, 2010.

[135] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, L. Van Gool, Robust tracking-by-detection using a detector confidence particle filter, in: Computer Vision, 2009 IEEE 12th International Conference on, IEEE, 2009, pp. 1515–1522.

[136] J. Xu, C. R. Shelton, Intrusion detection using continuous time bayesian networks, arXiv preprint arXiv:1401.3851.

[137] H. Altwaijry, Bayesian based intrusion detection system, in: IAENG Transactions on Engineering Technologies, Springer, 2013, pp. 29–44.

[138] X. Han, L. Xu, M. Ren, W. Gu, A naive bayesian network intrusion detection algorithm based on principal component analysis, in: Information Technology in Medicine and Education (ITME), 2015 7th International Conference on, IEEE, 2015, pp. 325–328.

[139] L. Scrucca, M. Fop, T. B. Murphy, A. E. Raftery, mclust 5: Clustering, classification and density estimation using gaussian finite mixture models, The R Journal 8 (1) (2016) 289.

[140] Z. Tan, A. Jamdagni, X. He, P. Nanda, R. P. Liu, A system for denial-of-service attack detection based on multivariate correlation analysis, IEEE transactions on parallel and distributed systems 25 (2) (2014) 447–456.

[141] W. Fan, N. Bouguila, D. Ziou, Unsupervised anomaly intrusion detection via localized bayesian feature selection, in: Data Mining (ICDM), 2011 IEEE 11th International Conference on, IEEE, 2011, pp. 1032–1037.

[142] N. Greggio, Learning anomalies in idss by means of multivariate finite mixture models, in: Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on, IEEE, 2013, pp. 251–258.

[143] C. Gruhl, B. Sick, A. Wacker, S. Tomforde, J. Hähner, A building block for awareness in technical systems: Online novelty detection and reaction with an application in intrusion detection, in: Awareness Science and Technology (iCAST), 2015 IEEE 7th International Conference on, IEEE, 2015, pp. 194–200.

[144] K. Singh, S. C. Guntuku, A. Thakur, C. Hota, Big data analytics framework for peer-to-peer botnet detection using random forests, Information Sciences 278 (2014) 488–497.

[145] C. Jirapummin, N. Wattanapongsakorn, P. Kanthamanon, Hybrid neural networks for intrusion detection system, in: Proc. of ITC–CSCC, 2002, pp. 928–931.

[146] K. Lee, J. Kim, K. H. Kwon, Y. Han, S. Kim, Ddos attack detection method using cluster analysis, Expert Systems with Applications 34 (3) (2008) 1659–1665.

[147] A. Jadhav, A. Jadhav, P. Jadhav, P. Kulkarni, A novel approach for the design of network intrusion detection system (nids), in: Sensor Network Security Technology and Privacy Communication System (SNS & PCS), 2013 International Conference on, IEEE, 2013, pp. 22–27.

[148] P. Saurabh, B. Verma, An efficient proactive artificial immune system based anomaly detection and prevention system, Expert Systems with Applications 60 (2016) 311–320.

[149] K. Ilgun, R. A. Kemmerer, P. A. Porras, State transition analysis: A rule-based intrusion detection approach, IEEE transactions on software engineering 21 (3) (1995) 181–199.

[150] P. Pudil, J. Novovičová, Novel methods for feature subset selection with respect to problem knowledge, in: Feature Extraction, Construction and Selection, Springer, 1998, pp. 101–116.

[151] S. Dubey, J. Dubey, Kbb: A hybrid method for intrusion detection, in: Computer, Communication and Control (IC4), 2015 International Conference on, IEEE, 2015, pp. 1–6.

[152] J. Pang, D. Liu, Y. Peng, X. Peng, Anomaly detection based on uncertainty fusion for univariate monitoring series, Measurement 95 (2017) 280–292.

[153] F. A. Narudin, A. Feizollah, N. B. Anuar, A. Gani, Evaluation of machine learning classifiers for mobile malware detection, Soft Computing 20 (1) (2016) 343–357.

[154] Y. Wang, Statistical Techniques for Network Security: Modern Statistically-Based Intrusion Detection and Protection: Modern Statistically-Based Intrusion Detection and Protection, IGI Global, 2008.

[155] P. A. Porras, A. Valdes, Live traffic analysis of tcp/ip gateways., in: NDSS, 1998.

[156] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, J. Tygar, Adversarial machine learning, in: Proceedings of the 4th ACM workshop on Security and artificial intelligence, ACM, 2011, pp. 43–58.

[157] G. A. Fink, B. Chappell, T. Turner, K. O'Donoghue, A metrics-based approach to intrusion detection system evaluation for distributed real-time systems, in: Parallel and Distributed Processing Symposium., Proceedings International, IPDPS 2002, Abstracts and CD-ROM, IEEE, 2001, pp. 8–pp.

[158] W.-C. Lin, S.-W. Ke, C.-F. Tsai, Cann: An intrusion detection system based on combining cluster centers and nearest neighbors, Knowledge-based systems 78 (2015) 13–21.

[159] M. Jabbar, R. Aluvalu, et al., Rfaode: A novel ensemble intrusion detection system, Procedia Computer Science 115 (2017) 226–234.

[160] W. Haider, J. Hu, J. Slay, B. Turnbull, Y. Xie, Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling, Journal of Network and Computer Applications 87 (2017) 185–192.

[161] K. A. Costa, L. A. Pereira, R. Y. Nakamura, C. R. Pereira, J. P. Papa, A. X. Falcão, A nature-inspired approach to speed up optimum-path forest clustering and its application to intrusion detection in computer networks, Information Sciences 294 (2015) 95–108.

[162] G. Creech, J. Hu, A semantic approach to host-based intrusion detection systems using contiguousand discontiguous system call patterns, IEEE Transactions on Computers 63 (4) (2014) 807–819.

[163] M. Hasan, T. Dean, F. T. Imam, F. Garcia, S. P. Leblanc, M. Zulkernine, A constraint-based intrusion detection system, in: Proceedings of the Fifth European Conference on the Engineering of Computer-Based Systems, ACM, 2017, p. 12.

[164] S. M. H. Bamakan, H. Wang, Y. Shi, Ramp loss k-support vector classification-regression; a robust and sparse multi-class approach to the intrusion detection problem, Knowledge-Based Systems 126 (2017) 113–126.

[165] B. Wang, Y. Zheng, W. Lou, Y. T. Hou, Ddos attack protection in the era of cloud computing and software-defined networking, Computer Networks 81 (2015) 308–319.

[166] B. Li, J. Springer, G. Bebis, M. H. Gunes, A survey of network flow applications, Journal of Network and Computer Applications 36 (2) (2013) 567–581.