IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 25 April 2022, accepted 21 June 2022, date of publication 29 June 2022, date of current version 7 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3187116

RESEARCH ARTICLE

Detecting Zero-Day Intrusion Attacks Using Semi-Supervised Machine Learning Approaches

INNOCENT MBONA AND JAN H. P. ELOFF
Department of Computer Science, University of Pretoria, Pretoria 0002, South Africa
Corresponding author: Innocent Mbona (u15256422@tuks.co.za)

The work of Innocent Mbona was supported in part by the University of Pretoria, and in part by the Bank Seta.

ABSTRACT Recently, network intrusion attacks, particularly new unknown attacks referred to as zero-day attacks, have become a global phenomenon. Zero-day network intrusion attacks constitute a frequent cybersecurity threat, as they seek to exploit the vulnerabilities of a network system. Previous studies have demonstrated that zero-day attacks can compromise a network for prolonged periods if network traffic analysis (NTA) is not performed thoroughly and efficiently. NTA plays a crucial role in supporting machine learning (ML) based network intrusion detection systems (NIDS) by monitoring and extracting meaningful information from network traffic data. Network traffic data constitute large volumes of data described by features such as destination-to-source packet count. It is important to use only those features that have a significant impact on the performance of an NIDS. The problem is that most existing ML models for NIDS employ features such as Internet protocol (IP) addresses that are redundant for detecting zero-day attacks and therefore negatively impact the performance of these ML models. The solution proposed in this study demonstrates that the law of anomalous numbers, famously known as Benford's law, is a viable technique that can effectively identify significant network features that are indicative of anomalous behaviour and can be used for detecting zero-day attacks. Finally, our study illustrates that semi-supervised ML approaches are effective for detecting zero-day attacks if significant features are optimally chosen. The experimental results demonstrate that one-class support vector machines achieved the best results (Matthews correlation coefficient of 74% and F_1 score of 85%) for detecting zero-day network attacks.

INDEX TERMS Benford's law, cybersecurity, significant features, network intrusion detection system, network traffic analysis, machine learning, zero-day attack.

I. INTRODUCTION

Wireless networks have become increasingly important given the rise of technologies such as the Internet of Everything (IoE) [1], [2]. Although wireless networks propel computers and humans to interact, such interactions result in high cybersecurity risk [1]–[3]. Recent reports on cybersecurity incidents indicate a continuous increase in cyberattacks, most notably zero-day network intrusion, data breaches, malware, and social engineering attacks [1]. Cybersecurity refers to the protection of computers, interconnected systems, electronic data, and wireless networks (among others) from threats to the confidentiality, integrity, and availability of systems [3], [4]. Wireless networks are vulnerable to cyber-attacks, also known as intrusions such as distributed denial-of-service (DDoS) attacks, as they do not have physical boundaries that may provide security and privacy [5]. For example, around October 2018 multiple web services, including Twitter, Spotify and GitHub, fell victim to a series of DDoS attacks [4]–[6]. This incident was one of the largest cyberattacks in history and resulted in devastating losses for web service companies. The vulnerabilities of wireless networks are often mitigated by solutions such as firewalls and network intrusion detection systems (NIDS), which are software located at a specific point within a wireless network to monitor network traffic and detect malicious activities [7].

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

An NIDS forms part of cybersecurity and plays a vital role in detecting and proactively neutralising attacks, such as denial-of-service (DoS) and web attacks [3], [5], [7].

NIDS are generally based on deep learning (DL) or machine learning (ML) techniques [8], [9]; in this study, we focus on the latter. ML-based NIDS, referred to as ML-NIDS, can further be categorised as supervised, unsupervised, or semi-supervised, depending on the availability of a labelled dataset to train the model [8]–[10]. Irrespective of an ML model type, the foremost steps in designing effective NIDS are data cleaning and feature selection [11], [12]. Feature selection is the process of identifying significant features and simultaneously removing redundant features from a dataset [11]–[13]. In this study, we used feature and attribute terms interchangeably without any impact, although the terms have slightly different meanings. Network traffic data comprise various attributes or features that describe network flows and host information [13]. However, some attributes, such as destination IP address, are redundant and can spoil the efficiency and accuracy of an ML model [13]. The selected features were used as input variables in the ML-NIDS design. Significant features can be defined as the subset of features that can affect the performance or computational cost of an ML model [11]–[13]. Feature selection is particularly important for network intrusion datasets, as they are typified by the high dimensions and disproportional sizes of benign and malicious network traffic [5]–[14]. Identifying significant features in high-dimensional and disproportional datasets can be computationally expensive [11]–[13].

A network traffic dataset is high-dimensional (having many features) and imbalanced (having disproportional sizes) [1]–[5]. Moreover, the volume and velocity aspects of a network traffic dataset cause conventional feature selection methods such as principal component analysis (PCA) to be computationally expensive [15]. A plethora of feature selection methods [14], [16], [17] for NIDS have been proposed thus far, with reference to network intrusion datasets. The challenge is that network traffic data are generated at high volumes and velocities, which inevitably increases the computational cost of these methods when analysing every network feature in real time to determine whether it is benign or malicious[14]–[17]. Therefore, there is a need for effective and efficient feature selection methods, particularly to detect zero-day attacks [17]–[19]. A zero-day network attack is defined as any new attack that seeks to exploit unknown or known vulnerabilities in a network system [17], [18]. Zero-day attacks are difficult to prevent because they can be executed in numerous ways, including brute force and botnet attacks [4]–[6].

State-of-the-art research on ML-NIDS can be grouped into signature-based, anomaly based, and hybrid-based detection [3], [5], [8]. Signature-based methods, also known as pattern-matching methods, are primarily used to detect specific network attacks [20]. For example, SNORT, an open-source network intrusion prevention system, is a well-known signature-based system [1]. Under signature-based methods, each network traffic is compared to a database of known attacks to determine whether network traffic is malicious or not [5]. Therefore, signature-based NIDS are effective in detecting specific attacks or known malicious activities, but are not as effective in detecting zero-day attacks or variants of known attacks [8], [20], [21]. In general, signature-based methods adopt supervised ML approaches that require large amounts of labelled data (i.e. normal and abnormal network traffic) to train the model [5]. Because obtaining large amounts of labelled data is difficult and time-consuming, this approach is usually not preferred in practice for detecting zero-day attacks [22]. An anomaly based NIDS tries to model normal (benign) network traffic and then deems any network activities that deviate from this expected behaviour, such as zero-day attacks, as anomalies [23]–[26]. Anomaly based NIDS are generally based on unsupervised ML approaches, and since they do not require labelled datasets to train a model, they can detect zero-day attacks [5]. However, large amounts of data are still required to train a mode [9]. A third option is to use a hybrid-based NIDS that combines signature-based and anomaly-based detection to detect known and unknown attacks, and they are based on semi-supervised ML approaches [5]. Semi-supervised learning approaches are particularly appealing to NIDS given their ability to learn from small sets of labelled data and large amounts of unlabelled data [27]. Obtaining unlabelled data is easier than obtaining labelled data.

In this study, we adopted semi-supervised ML approaches to detect zero-day network intrusion attacks because of the challenge of obtaining large amounts of labelled data. In Figure 1, we highlight our overall approach to detecting zero-day network intrusion attacks.

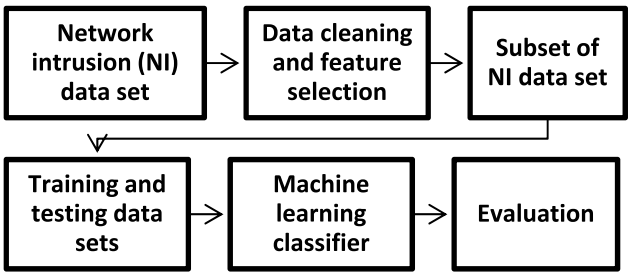


FIGURE 1. Our approach in detecting Zero-day network intrusion attacks.

Figure 1 shows that we first sourced network intrusion datasets and then performed data cleaning and feature selection. We obtained a subset of network intrusion datasets that were further split into training and testing datasets. Finally, we implemented various semi-supervised ML approaches to detect zero-day attacks and evaluated the performance of each ML model.

A. RESEARCH GOALS

- Review and discuss current methods used to detect or prevent zero-day attacks.

• Determine the network traffic features that should be monitored to proactively detect zero-day attacks.
• To investigate whether semi-supervised ML models can effectively detect zero-day attacks.

The rest of the paper is organised as follows: Section II discusses related previous studies. Section III contains descriptions of Benford's law and semi-supervised ML used in this study. Section IV contains descriptions of all datasets used in this study. Section V contains feature selection and semi-supervised ML results for detecting zero-day attacks, as well as discussions. Section VI concludes the study and suggests plans for future research.

II. RELATED STUDIES

In Section II, we sample key state-of-the-art studies related to zero-day attacks, feature selection on network intrusion datasets, and ML approaches for detecting zero-day network intrusion attacks.

A. NETWORK INTRUSION DETECTION SYTEMS

Pu *et al.* [28] proposed a hybrid unsupervised anomaly based NIDS to detect zero-day network intrusion attacks. Their approach was based on the combination of a subspace clustering (SSC) and one-class support vector machine (OCSVM) models, and they evaluated this hybrid model using the NSL-KDD network intrusion dataset. The SSC model was used to group unlabelled data, and the OCSVM was used to train the model using only one class (i.e. benign network traffic). Pu *et al.* [28] applied the F-test feature selection method to identify significant features in the NSL-KDD dataset. Although their hybrid model achieved a detection rate of 99% for zero-day attacks, the authors did not specify which significant features were used to achieve these results. Anomaly based approaches detect zero-day attacks by modelling normal (benign) network traffic such that traffic that deviates from this expected behaviour is deemed an anomaly (including zero-day attacks) [9]– [28]. As much as anomaly based NIDS are appealing for detecting zero-day attacks, they generally suffer from a high false-positive[1] or false-negative[2] rate (compared to signature-based methods), owing to a lack of definitive training datasets and the imbalanced nature of network traffic data [20], [22], [24].

Vahdani *et al.* [18] implemented a two-phase unsupervised ML approach to detect zero-day attacks. The first phase detects zero-day attacks by monitoring network flows using a self-adaptable technique. The second phase uses the DBSCAN clustering technique to detect similar attacks (e.g. botmaster). The authors evaluated their methodology using the DAPRA and ISCX datasets and achieved an accuracy of 98%. Chiba *et al.* [29] implemented an anomaly based NIDS based on a backpropagation neural network (BPNN), to detect zero-day network attacks. The BPNN was evaluated using the KDD CUP'99 dataset. The authors implemented two feature selection methods – one of which was the Kolmogorov-Smirnov correlation-based filter method, and identified 12 significant features from a total of 41 features. The information gain method identified 17 significant features, while 34 numeric features of the KDD CUP'99 data sets were considered. In all three sets of significant features, numeric features such as destination_host_count were observed to be significant, in contrast to non-numeric features such as the protocol type. Because protocol-type features can be either an Internet control message protocol (ICMP), user datagram protocol (UDP), or transmission control protocol (TCP), they display indistinguishable patterns for both benign and malicious network traffic, thus rendering this type of feature redundant for the classification of benign and zero-day attacks.

Zavrak *et al.* [22] implemented autoencoder DL methods based on semi-supervised learning to detect zero-day intrusion attacks. They used the CICIDS2017 dataset, that contains benign and 11 malicious attacks. The variational autoencoder (VAE), autoencoder (AE), and OCSVM achieved area under the curve (AUC) results of 76%, 74%, and 66%, respectively, for zero-day attacks. Interestingly, these models achieve better results in detecting specific attacks. For example, OCSVM achieved 98.5% AUC for heartbleed attacks, 90% for infiltration using VAE, and 83% for DoS hulk using AE. These results demonstrate that both ML and DL approaches can perform well in the detection of specific known malicious network traffic; however, their performance may decrease for zero-day attacks. Hindy *et al.* [24] likewise adopted the AE approach to detect zero-day attacks using the CICIDS2017 and NSL-KDD datasets, where they achieved an accuracy of 99% and 98% on the NSL-KDD and CICIDS2017 datasets, respectively.

Abdalgawad *et al.* [30] demonstrated that DL methods including adversarial autoencoders (AAE) and bidirectional adversarial networks (BiGAN) are effective for detecting zero-day network attacks. These methods were evaluated on the IoT-23 dataset, which comprises of 15 network attacks and 19 network traffic features. Out of the 19 features, the authors deemed only eight to be significant, which included the transaction protocol (ICMP, UDP, and TCP). This contradicts the findings of Chiba *et al.* [29] and Zoppi *et al.* [9] for different datasets. Zoppi *et al.* [9] implemented various unsupervised ML models to detect zero-day intrusion attacks using an SDN20 dataset. This dataset contains benign and five types of malicious network traffic that are described by 85 features. The authors identified 12 significant features using the rapid evaluation of anomaly detection algorithms, all of which were numeric, such as average packet size. The bagging ensemble classifier produced the best results, with an F_1 score of 96% and a Matthews correlation coefficient (MCC) of 68%. Blaise *et al.* [25] proposed an unsupervised anomaly based approach called split-and-merge for detecting zero-day network intrusion attacks. They first used the F-test feature selection method on the MAWI and UCSD datasets;

[1]False positive rate is the number of benign network traffic instances that are incorrectly classified as malicious.

[2]False negative rate is the number of malicious network traffic instances that are incorrectly classified as benign.

TABLE 1. Summary of well-known methods for detecting network Zero-day attacks.

Authors	Data set(s)	Performance	Approach	Classifier(s) used
Pu et al. [28]	NSL-KDD	Detection rate – 99%	Unsupervised learning	Sub-space clustering and (OCSVM)
Chiba et al. [29]	KDD CUP'99	F_1 score – 99%	Unsupervised learning	Backpropagation neural network
Zoppi et al. [9]	SDN20	F_1 score – 96%	Unsupervised learning	Bagging ensemble methods
Taher et al. [12]	NSL-KDD	Accuracy – 94%	Supervised learning	Artificial neural network
Abri et al. [8]	Meraz'18	Accuracy – 99%	Supervised learning	Random forest
Vahdani et al. [18]	DARPA & ISCX	Precision – 98%	Unsupervised learning	DBSCAN clustering
Hindy et al. [24]	CICIDS2017 & NSL-KDD	Accuracy – 99%	Deep learning	Artificial neural network
Abdalgawad et al. [30]	IoT-23	F_1 score – 85%	Deep learning	AAE and BiGAN
Blaise et al. [25]	MAWI & UCSD Network	True Positive Rate – 86%	Unsupervised learning	Split-and-merge
Pallaprolu et al. [32]	KDD CUP'99	Accuracy – 98%	Supervised learning	SLN and dynamic semantic graph generation
Duong et al. [27]	NSL-KDD	Recall – 89%	Semi-supervised learning	Mahanalobis distance PCA
Zhu et al. [31]	NSL-KDD	Accuracy – 80%	Semi-supervised learning	Decision tree and k-NN
Zavrak et al. [22]	CICIDS2017	AUC – 76%	Deep learning	Variational autoencoder

their methodology achieved 79% and 86% true positive rates on the MAWI and UCSD datasets, respectively.

Taher *et al.* [12] implemented artificial neural networks (ANN) based on supervised anomaly based ML to detect zero-day attacks. Furthermore, they used the wrapper feature selection method on the NSL-KDD dataset (one benign and four malicious attacks) to identify the significant features. The wrapper method identified 17 significant features from a total of 41. Taher *et al.* [12] did not specify the significant features; however, in a supervised setting, they used 20% of the NSL-KDD dataset, which has 25 191 labelled instances. The remaining 80% were used to test the model. Although the ANN model produced a high (94%) detection rate, the limitation of this study stems from the relatively small dataset used for supervised learning. Duong *et al.* [27] used the NSL-KDD dataset to propose a semi-supervised ML approach to detect zero-day attacks. Their approach was based on a modified Mohanalobis distance principal component analysis (M-PCA) and K-means clustering model. The k-means clustering model was used in the training phase to remove outliers from the training dataset. Ideally, the training dataset for the semi-supervised approach should consist of only one class (i.e. benign) of network traffic. M-PCA was used in the testing phase to evaluate the model. Duong *et al.* [27] identified six significant features, including the protocol type, using an unspecified feature selection method. We note a discrepancy in the number of significant features identified by Duong *et al.* [27] and Taher *et al.* [12] on the same dataset (NSL-KDD). The M-PCA method achieved an accuracy of 91% for the detection rate. Zhu *et al.* [31] proposed the use of an ensemble semi-supervised machine-learning approach to detect zero-day network attacks. Their ensemble approach is based on a decision tree and k-nearest neighbour (kNN) algorithm. The authors achieved an accuracy of 80% by using the NSL-KDD dataset to detect zero-day attacks. However, Zhu *et al.* [31] did not specify the significant features used to obtain these results.

Pallaprolu *et al.* [32] proposed that a combination of semantic link networks (SLN) and dynamic semantic graph generation can be used to detect zero-day network intrusion attacks. They used the KDD CUP'99 dataset to evaluate their methodology, and implemented the minimum redundancy maximum relevance (MRMR) feature selection method. The MRMR feature selection method identified 25 significant features consistent with those identified by Chiba *et al.* [29], as discussed above. The approach adopted by Pallaprolu *et al.* [32] achieved an accuracy of 98% for detecting zero-day intrusion attacks. Abri *et al.* [8] evaluated various types of ML and DL classifiers to detect zero-day attacks and used the Meraz'18 data set to evaluate their models. This dataset contains benign and malicious network traffic described by 55 features. The authors did not apply feature selection; instead, they manually removed three features. Having many features in the training dataset can increase the computational cost of a model and cause overfitting problems [11]–[13]. The Gaussian naïve bayes ML model performed poorly, with an accuracy of 46.31%, whereas the random forest model achieved the best results with an accuracy of 99.51%. DL approaches, such as the multilayer perceptron, also achieved an accuracy of 99.25%. In Table 1, we summarise the existing approaches, datasets, performances, and methods used to detect zero-day attacks.

TABLE 2. Benford's results using the total length of backward packets feature from the CICDDoS2019 data set.

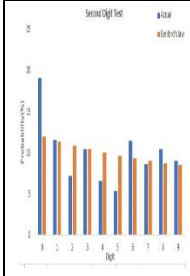
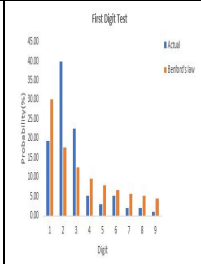
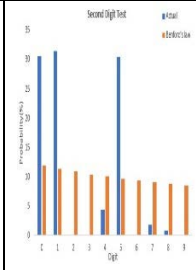
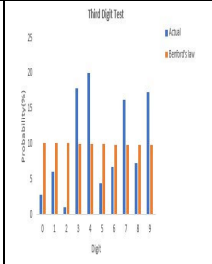
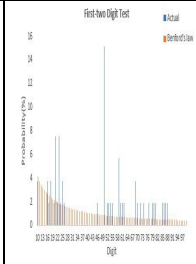
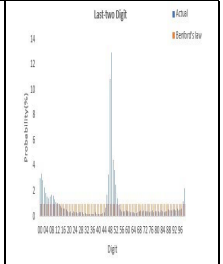
Benign (Do not reject H_0) SDT	DDoS_DNS (reject H_0) FDT	DDoS_DNS (reject H_0) SDT	DDoS_DNS (reject H_0) TDT	DDoS_DNS (reject H_0) F2DT	DDoS_DNS (reject H_0) L2DT
					

TABLE 3. Summary of the UNSW-NB15 data set and description of cyberattacks [13].

Network traffic	Description	No. of instances
Benign	Normal network traffic.	56000
DoS	The attacker attempts to deny an authorised person access to a network or device.	12264
Generic	The attacker attempts to collide block-cyphers using the hash function.	40000
Worm	The attacker uses malware that can replicate itself rapidly and spread across devices in a network.	130
Shellcode	This is a malware code used by an attacker that attempts to control a compromised device.	1133
Reconnaissance	The attacker attempts to illegally collect network information so as to bypass the security controls.	10491
Fuzzing	The attacker attempts to overload a network by feeding it massive random data.	18184
Analysis	The attacker analyses a network structure to determine network ports and nodes that can be exploited.	2000
Backdoor	The attacker attempts to bypass a customary network security control.	1746
Exploit	This is a malware code that exploits a vulnerability in a network that intends to cause the network to malfunction.	33393

We benchmarked the proposed approach against these results.

B. FEATURE SELECTION METHODS

The ML-NIDS used for detecting zero-day attacks relies on input data, referred to as features [14]–[21]. The performance of ML-NIDS methods can be improved using significant features that are indicative of anomalous behaviour between benign and zero-day network traffic [16], [17]. NIDS utilises network traffic analysis (NTA) methods to capture and analyse network traffic data to detect threats, such as zero-day attacks. The main challenge that currently faces an NIDS is to extract significant features effortlessly in real time. A network traffic feature is deemed significant if it can distinguish between benign and malicious (zero-day) network traffic [33], [34]. Our study aimed to address the challenge of seamlessly extracting significant features in an attempt to detect unknown malicious attacks. In our approach toward identifying significant features to detect zero-day attacks, we used well-known publicly available datasets and opted to use the original features found in these datasets. We did not derive new features.

Moustafa *et al.* [13] worked at the Cyber Range Lab of UNSW Canberra and created the UNSW-NB15 (see Table 3) and KDD99 datasets, which consisted of nine cyberattacks and benign network traffic. Although the UNSW-NB15 dataset contains a total of 47 features, in Table 4 we highlight only numeric features, as previous studies [9], [29], [32] demonstrate the importance of numeric features to better describe network traffic behaviour. The 47 features in this dataset were grouped into six distinct groups: flow features, basic features, content features, time features, additional features, and labelled features. Each group of features contains certain information about the network; for example, time features contain information about interpacket arrival times. Moustafa *et al.* [13] applied the association rule mining (ARM) method to identify 27 significant numerical features.

TABLE 4. Highlights numerical features of the UNSW-NB15 data set.

No.	NTA feature	No.	NTA feature
1	Row total duration	22	The time between the SYN and the SYN_ACK packets
2	Source to destination packet count	23	The time between the SYN_ACK and the ACK packets
3	Destination to source packet count	24	Mean of the packet size transmitted by the srcip
4	Source to destination bytes	25	Mean of the packet size transmitted by the dstip
5	Destination to source bytes	26	The connection of http request/response transaction
6	Rate	27	The content size of the data transferred from http
7	Source to destination time to live	28	Number of rows of the same service and srcip
8	Destination to source time to live	29	Number of each state according to values of sttl and dttl
9	Source bits per second	30	Number of rows of the srcip
10	Destination bits per second	31	Number of rows of the same srcip and the dsport
11	Source packets retransmitted or dropped	32	Number of rows of the same dstip and the sport
12	Destination packets retransmitted or dropped	33	Number of rows of the same srcip and the dstip
13	Source interpacket arrival time (mSec)	34	is_ftp_login
14	Destination interpacket arrival time (mSec)	35	Number of flows that have a command in ftp session
15	Source jitter	36	Number of methods such as Get and Post in http service
16	Destination jitter	37	Number of rows of the same dstip
17	Source TCP window advertisement value	38	Number of rows of the same service and dstip
18	Source TCP base sequence number	39	If srcip = dstip and sport = dsport assign 1, else 0
19	Destination TCP base sequence number		
20	Destination TCP window advertisement value		
21	Setup round-trip time, the sum of 'synack' and 'ackdat'		

TABLE 5. Summary of instances of various DDoS attacks from the CICDDoS2019 data set [35].

Network traffic	Description	No. of instances
Benign	Normal network traffic.	26719
DDoS_DNS	The attacker uses DNS to execute DDoS.	1046567
DDoS_LDAP	The attacker uses LDAP to execute DDoS.	1047795
DDoS_MSSQL	The attacker uses MSSQL to execute DDoS.	1047561
DDoS_NetBIOS	The attacker uses NetBIOS to execute DDoS.	1047706
DDoS_NTP	The attacker uses NTP to execute DDoS.	1035009
DDoS_SNMP	The attacker uses SNMP to execute DDoS.	1047663
DDoS_SSDP	The attacker uses SSDP to execute DDoS.	1048271
DDoS_UDP	The attacker uses UDP to execute DDoS.	1047441
DDoS_SYN	The attacker uses SYN to execute DDoS.	1048228
DDoS_TFTP	The attacker uses TFTP to execute DDoS.	1047402
DDoS_UDPLag / Web	The attacker uses UDPLag or Web to execute DDoS.	366900

The ARM method identifies significant features based on an iterative process for generating rules of support and confidence [13]. This iterative process can be time-consuming, and in Section V, we demonstrate that a straightforward Benford's law method identifies similar significant features.

Sharafaldin *et al.* [35] proposed a new taxonomy for different types of DDoS attacks and generated a dataset named CICDDoS2019 (see Table 5). This dataset covers 11 different types of DDoS attacks, including DNS-DDoS and SYN-DDoS attacks, and 80 network features (see Table 6). When we investigated these features in more detail, we observed that most were based on distinct features but different statistical measures. For example, the forward packet length attribute has various features, such as forward packet length

TABLE 6. Highlights numerical features of the CICDDoS2019 data set.

No.	NTA feature	No.	NTA feature	No.	NTA feature	No.	NTA feature
1	Source Port	22	Flow IAT Min	43	Packet Length Mean	64	Bwd Avg Bulk Rate
2	Destination Port	23	Fwd IAT Total	44	Packet Length Std	65	Subflow Fwd Packets
3	Protocol	24	Fwd IAT Mean	45	Packet Length Variance	66	Subflow Fwd Bytes
4	Flow Duration	25	Fwd IAT Std	46	FIN Flag Count	67	Subflow Bwd Packets
5	Total Fwd Packets	26	Fwd IAT Max	47	SYN Flag Count	68	Subflow Bwd Bytes
6	Total Bwd Packets	27	Fwd IAT Min	48	RST Flag Count	69	Init_Win_bytes_forward
7	Total Length of Fwd Packets	28	Bwd IAT Total	49	PSH Flag Count	70	Init_Win_bytes_backward
8	Total Length of Bwd Packets	29	Bwd IAT Mean	50	ACK Flag Count	71	act_data_pkt_fwd
9	Fwd Packet Length Max	30	Bwd IAT Std	51	URG Flag Count	72	min_seg_size_forward
10	Fwd Packet Length Min	31	Bwd IAT Max	52	CWE Flag Count	73	Active Mean
11	Fwd Packet Length Mean	32	Bwd IAT Min	53	ECE Flag Count	74	Active Std
12	Fwd Packet Length Std	33	Fwd PSH Flags	54	Down/Up Ratio	75	Active Max
13	Bwd Packet Length Max	34	Bwd PSH Flags	55	Average Packet Size	76	Active Min
14	Bwd Packet Length Min	35	Fwd URG Flags	56	Avg Fwd Segment Size	77	Idle Mean
15	Bwd Packet Length Mean	36	Bwd URG Flags	57	Avg Bwd Segment Size	78	Idle Std
16	Bwd Packet Length Std	37	Fwd Header Length	58	Fwd Header Length.1	79	Idle Max
17	Flow Bytes/s	38	Bwd Header Length	59	Fwd Avg Bytes/Bulk	80	Idle Min
18	Flow Packets/s	39	Fwd Packets/s	60	Fwd Avg Packets/Bulk		
19	Flow IAT Mean	40	Bwd Packets/s	61	Fwd Avg Bulk Rate		
20	Flow IAT Std	41	Min Packet Length	62	Bwd Avg Bytes/Bulk		
21	Flow IAT Max	42	Max Packet Length	63	Bwd Avg Packets/Bulk		

mean/minimum/maximum (etc.). This makes this dataset more complete because it uses different statistical measures to describe attributes. Sharafaldin *et al.* [35] applied random forest regressor (RFR) feature selection method to identify significant features for each DDoS type. A well-documented limitation of the RFR method is that correlated features will have similar importance; however, their importance would differ if the tree was designed without the correlated counterparty feature [36], [37]. In this study, we demonstrate that a simple Benford's law method can effectively identify significant features to distinguish between benign and zero-day DDoS-type attacks.

Ullah *et al.* [38] examined seven cyberattacks, inter alia spoofing and host port, in the Internet of Things (IoT) networks and proposed a data set named IoTID20 (Table 7). This dataset contained 79 features (see Table 8), and significant features were identified using the recursive feature elimination (RFE) method. The RFE method is computationally expensive if a dataset contains a large number of features [38]. The features used by Ullah *et al.* [38] on their dataset were similar to those used by Sharafaldin *et al.* [35], although the attacks considered on these datasets were completely different. Montazerishatoori *et al.* [39] proposed a new taxonomy for DNS over HTTPS (DoH) tunnel traffic and proposed a dataset named CIRA-CIC-DoHBrw20 (see Table 9). This dataset contains 31 numerical features (see Table 10) that are categorised as either time series, payload inspection, or statistical features. The authors also used the RFR feature selection method, as discussed earlier. This study aims to demonstrate that Benford's law can effectively differentiate between benign and malicious DoH tunnel traffic.

TABLE 7. Summary of instances of various cyberattacks from the IoTID20 data set [38].

Network traffic	Description	No. of instances
Benign	Normal network traffic.	40073
SYN Flooding	The attacker attempts to cause a network malfunction by setting up multiple half-open connections without finalising them.	59391
UDP Flooding	The attacker attempts to overwhelm random ports in a network so that they will not be available to legitimate users.	183554
HTTP Flooding	The attacker attempts to manipulate the normal HTTP with a malicious one to attack a web server.	55818
ACK Flooding	The attacker attempts to overwhelm the network server with ACK packets.	55124
Host Brute force	The attacker attempts to guess the password of a host by submitting numerous passwords.	121181
ARP Spoofing	The attacker sends manipulated Address Resolution Protocol (ARP) to illegally connect over a network.	35377
Scan host port	The attacker attempts to gather information about the device by issuing multiple hotspots.	22192
Scan port OS	The attacker observes a network to figure out open ports that can be exploited.	53073

TABLE 8. Numerical features of the IoTID20 data set.

No.	NTA feature	No.	NTA feature	No.	NTA feature	No.	NTA feature
1	Source_Port	**22**	Flow_IAT_Min	**43**	Pkt_Len_Mean	**64**	Subflow_Fwd_Pkts
2	Destination_Port	**23**	Fwd_IAT_Tot	**44**	Pkt_Len_Std	**65**	Subflow_Fwd_Byts
3	Protocol	**24**	Fwd_IAT_Mean	**45**	Pkt_Len_Var	**66**	Subflow_Bwd_Pkts
4	Flow_Duration	**25**	Fwd_IAT_Std	**46**	FIN_Flag_Cnt	**67**	Subflow_Bwd_Byts
5	Total_Fwd_Pkts	**26**	Fwd_IAT_Max	**47**	SYN_Flag_Cnt	**68**	Init_Fwd_Win_Byts
6	Total_Bwd_Pkts	**27**	Fwd_IAT_Min	**48**	RST_Flag_Cnt	**69**	Init_Bwd_Win_Byts
7	Total_Length_Fwd_Pkts	**28**	Bwd_IAT_Tot	**49**	PSH_Flag_Cnt	**70**	Fwd_Act_Data_Pkts
8	Total_Length_Bwd_Pkts	**29**	Bwd_IAT_Mean	**50**	ACK_Flag_Cnt	**71**	Fwd_Seg_Size_Min
9	Fwd_Pkt_Len_Max	**30**	Bwd_IAT_Std	**51**	URG_Flag_Cnt	**72**	Active_Mean
10	Fwd_Pkt_Len_Min	**31**	Bwd_IAT_Max	**52**	CWE_Flag_Count	**73**	Active_Std
11	Fwd_Pkt_Len_Mean	**32**	Bwd_IAT_Min	**53**	ECE_Flag_Cnt	**74**	Active_Max
12	Fwd_Pkt_Len_Std	**33**	Fwd_PSH_Flags	**54**	Down/Up_Ratio	**75**	Active_Min
13	Bwd_Pkt_Len_Max	**34**	Bwd_PSH_Flags	**55**	Pkt_Size_Avg	**76**	Idle_Mean
14	Bwd_Pkt_Len_Min	**35**	Fwd_URG_Flags	**56**	Fwd_Seg_Size_Avg	**77**	Idle_Std
15	Bwd_Pkt_Len_Mean	**36**	Bwd_URG_Flags	**57**	Bwd_Seg_Size_Avg	**78**	Idle_Max
16	Bwd_Pkt_Len_Std	**37**	Fwd_Header_Len	**58**	Fwd_Byts/b_Avg	**79**	Idle_Min
17	Flow_Byts/s	**38**	Bwd_Header_Len	**59**	Fwd_Pkts/b_Avg		
18	Flow_Pkts/s	**39**	Fwd_Pkts/s	**60**	Fwd_Blk_Rate_Avg		
19	Flow_IAT_Mean	**40**	Bwd_Pkts/s	**61**	Bwd_Byts/b_Avg		
20	Flow_IAT_Std	**41**	Pkt_Len_Min	**62**	Bwd_Pkts/b_Avg		
21	Flow_IAT_Max	**42**	Pkt_Len_Max	**63**	Bwd_Blk_Rate_Avg		

Other notable previous studies on feature selection for network intrusion datasets are highlighted below. Yulianto et al [7] applied PCA, the synthetic minority oversampling technique (SMOTE), and ensemble feature selection (EFS) to the CIC-IDS-2017 dataset. In their experiments, PCA was the best performing method, although it is known to be computationally expensive. Kurniabudi et al [16] identified significant features of the CIC-IDS-2017 dataset using the information gain (IG) method and applied various ML methods to detect malicious attacks. The features of the CIC-IDS-2017 datasets are very similar to those detected by Ullah *et al.* [38] and Sharafaldin *et al.* [35].

TABLE 9. Summary of instances of DoH, non-DoH, benign and malicious network traffic from [39].

Network traffic	Description	No. of instances
Non-DoH	Non-DoH network traffic	897493
DoH	DoH network traffic	269643
Benign	Benign network traffic	19807
Malicious	Malicious network traffic	249836

Abdulhammed *et al.* [14] also applied PCA to the CIC-IDS-2017 dataset to identify significant features.

Our previous study [40] addressed the limitations of existing feature selection methods by proposing a novel method called Benford's law. Benford's law has been shown to detect network attacks such as DoS attacks [41] and portscan attacks [42]. Therefore, we propose using it as a feature selection method to overcome challenges that include class imbalance, high dimensions, and computational cost. We used the CIC-IDS2017 and CSE-CIC-IDS2018 data sets to demonstrate the effectiveness of the Benford's law method. Furthermore, significant features identified by Benford's law with respect to each attack were benchmarked against various feature selection methods, such as information gain, PCA, SMOTE, ensemble feature selection, chi-squared, XGBoost, CatBoost, lightGBM, and random forest. In this study, we further investigated whether Benford's law can identify significant features to distinguish between benign and zero-day attacks.

III. BACKGROUND

In this section, we provide the details of Benford's law and the semi-supervised ML models used in this study.

A. BENFORD's LAW (BL)

Benford's law (BL), as a feature selection method for detecting network traffic features, has recently received increased attention and has been applied to various application domain areas such as forensic auditing [33], fraud detection [43], and network intrusion [33]–[41]. Benford's law states that in naturally occurring systems, such as network systems, the distribution of the leading digits is non-uniform [44], [45]. Network systems are perceived as natural because the data collected from a network (such as packet sizes) represent real user behaviour [13]–[38]. NTA data mainly consist of two feature types: functional and data feature types. An example of the former is the protocol type, whereas an example of the latter is the flow duration. In many studies [1]–[46], numerical network aspects, such as packet sizes (packet-based), have been shown to be more effective than functional features, such as protocol type, in analysing benign and malicious network traffic. Therefore, our study aims to investigate data type features and, in particular, numerical data subtype features of network traffic that are indicative of anomalous behaviour between benign and zero-day network traffic. The BL distributions are formulated based on the following properties:

TABLE 10. Numerical features of the CIRA-CIC-DoHBrw-2020 data set. The features in Table 10 are mainly based on flow, packet, and time attributes.

No.	NTA feature	No.	NTA feature
1	SourcePort	22	PacketTimeSkewFromMode
2	DestinationPort	23	PacketTimeCoefficientofVariation
3	Duration	24	ResponseTimeTimeVariance
4	FlowBytesSent	25	ResponseTimeTimeStandardDeviation
5	FlowSentRate	26	ResponseTimeTimeMean
6	FlowBytesReceived	27	ResponseTimeTimeMedian
7	FlowReceivedRate	28	ResponseTimeTimeMode
8	PacketLengthVariance	29	ResponseTimeTimeSkewFromMedian
9	PacketLengthStandardDeviation	30	ResponseTimeTimeSkewFromMode
10	PacketLengthMean	31	ResponseTimeTimeCoefficientofVariation
11	PacketLengthMedian		
12	PacketLengthMode		
13	PacketLengthSkewFromMedian		
14	PacketLengthSkewFromMode		
15	PacketLengthCoefficientofVariation		
16	PacketTimeVariance		
17	PacketTimeStandardDeviation		
18	PacketTimeMean		
19	PacketTimeMedian		
20	PacketTimeMode		
21	PacketTimeSkewFromMedian		

Property 1

Let D be a positive real number on $(\Omega, f, \mathbb{P})$*. The logarithmic density function for the first leading digit is given by:*

$$\mathbb{P}(\mathrm{D} = \mathrm{d}) = \log_{\mathrm{b}}(1+\frac{1}{\mathrm{d}}) \tag{1}$$

where b = 10 and $d \in \{1,2,..9\}$.

Property 2

The logarithmic joint density of the first leading digits $\mathrm{D}_1, \mathrm{D}_2, .., \mathrm{D}_\mathrm{k}$ *on* $(\Omega, f, \mathbb{P})$ *for every* $\mathrm{k} \in \mathrm{N}$;

$$\mathbb{P}(\mathrm{D}_1 = \mathrm{d}_1, .., \mathrm{D}_\mathrm{k} = \mathrm{d}_\mathrm{k}) = \log_{10}(1+(\sum_{\mathrm{i}=1}^{\mathrm{k}} 10^{\mathrm{k}-\mathrm{i}}\mathrm{d}_\mathrm{i})^{-1}) \tag{2}$$

where $\mathrm{d}_1 \in \{1, 2, \ldots 9\}$ and all other $\mathrm{d}_\mathrm{j} \in \{0, 1, \ldots 9\}$.

There are five main Benford's law distribution tests: the first digit test (FDT), second digit test (SDT), first two-digit test (F2DT), third digit test (TDT), and last two-digit test (L2DT). The FDT, F2DT, and L2DT distributions are computed using Equation (1), and the remaining distributions are computed using Equation (2). The examples below demonstrate the computation of Benford's law of probabilities for the five distributions.

For each network feature in the datasets used in this study, we compared the Benford's law distribution with the actual distribution (benign and zero-day). We used various goodness-of-fit measures that included the Kolmogorov-Smirnov and Pearson chi-squared tests (see [33]–[44] for more details) and formulated the goodness-of-fit test as follows:

Null hypothesis (H_0) = a distribution that obeys Benford's law.

The alternative hypothesis (H_1) = a distribution that violates Benford's law.

If p-value < 0.05, we reject H_0, else we cannot reject H_0.

A network traffic feature is deemed significant if it obeys Benford's law on benign network traffic while simultaneously violating Benford's law on zero-day network traffic. We considered five key Benford's law tests, namely FDT, SDT, F2DT, TDT and L2DT. For example, let us consider the total length of the backward packet features from the CICDDoS2019 dataset.

Table 2 shows the application and results of BL using the total length of backward packet feature from the CICDDoS2019 dataset. In this case, this feature obeys Benford's law on benign network traffic on the SDT but simultaneously violates BL on the remaining Benford's law tests. Therefore, this was deemed to be a significant feature. Further results are presented in Appendix A.

B. GAUSSIAN MIXTURE MODEL

The multivariate Gaussian mixture model (GMM) is based on the Gaussian distribution which was established in the 80's by German mathematician Carl Friedrich Gauss. Over the years, GMM has been widely used to detect anomalies in continuous data systems, such as computer networks [49]. The GMM assumes that "normal" data points are grouped together, while anomalies such as zero-day network traffic deviate from the normal points. Given the training set $\{x_j^1, x_j^2, \ldots, x_j^n\}$, we modelled the mean (μ), variance $(\sum)$, and probability $(\mathbb{P})$ as follows:

(i) Fit model parameters

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_j^i \text{ and } \sum = \frac{1}{n}\sum_{i=1}^{n}(x_j^i - \mu)(x_j^i - \mu)^T$$

(ii) Given a new unlabelled data point $x_{unlabelled}$, compute

$$\mathbb{P}(x_{unlabeled}) = \frac{1}{(2)^{\frac{n}{2}}|\sum|^{\frac{1}{2}}}\exp(-\frac{1}{2}\left(x^i - \mu\right)^{T^{-1}}(x^i - \mu))$$

(iii) Predict;

$$Predict = \begin{cases} attack & \text{if } \mathbb{P}(x_{unlabelled}) < \varepsilon \\ normal & \text{otherwise} \end{cases}$$

where n is the number of samples and $\sum$ is n-dimensional covariance matrix, the threshold ε is determined automatically on the training dataset [49].

C. ONE CLASS SUPPORT VECTOR MACHINE

One-class support vector machines (OCSVM) were extended from standard SVMs originally introduced by Vapnik *et al.* [50], [51]. The objective of the OCSVM is to train a classifier using only one class (e.g. benign network traffic) such that anomalies (e.g. zero-day network traffic) can be detected. The OCSVM is formulated as follows:

(i) Given a training dataset, $Z = \{z_1, z_2, .., z_N\}$ of one class and let $\phi : Z \longrightarrow G$ be a kernel map which maps training data points to another space G. Then, to separate datasets from the origin, one must solve the following quadratic programming equation:

$$\min_{\in G, \xi_i, b \in \mathbb{R}} w \in \left\{\frac{1}{2}||w||^2 + \frac{1}{\nu N}\sum_{i=1}^{N}(\xi_i - b)\right\} \tag{3}$$

Subject to

$$\nu \in (0, 1], \xi_i \geq 0, \quad \forall_i = 1, 2, \ldots, N \text{ and } (w.\phi(z_i)) \geq b - \xi_i, \forall_i = 1, 2, \ldots, N \tag{4}$$

where ξ_i is a non-zero slack and ν is the ratio of anomalies in the training dataset, which we set to $\nu = 0.1$ for our case. Finally, the decision boundary function $f(z)$ becomes

$$f(z) = sign\{(w.\phi(z)) - b\}.$$

D. LABEL PROPAGATION

The label propagation (LP) algorithm was introduced by Zhou *et al.* [51]. The intuition behind the LP algorithm is that in the training dataset, points that are close to each other will have the same class label, and data points further to this

point will have their own similar label, that is, anomalies. Given N labelled data points and Munlabeled data points, denoted by $y= 0$. Let $G = \{V, E\}$ represent a graph with every vertex (E) comprising labels $V= \{+1, -1, 0\}$, where the edge is based on the affinity matrix W. The input features are denoted as X. The LP algorithm assumes that two nodes are connected if they are "similar"; therefore, unlabelled data points can be labelled by propagating the labelled data points until convergence is achieved. The LP algorithm is formulated as follows:

Given a dataset with N labeled and M unlabeled data points, i.e.,

$$\begin{cases} X = (X_L, X_U) \text{ where } X \varepsilon \mathbb{R} \\ Y = (Y_L, Y_U) \text{ where } Y \in \{-1, +1, 0\} \end{cases}$$

1) Using the k-nearest neighbors (kNN), compute the affinity matri W.
2) Compute the degree matri

$$D = diag(\left|\sum_j W_{ij}\right| \forall i= 1, 2, .., N + M)$$

3) Le $Y^{(0)} = Y$
4) Define $Y_L= \{y_0, y_1, \ldots, y_N\}$
5) Iterate until convergence is achieve

$$\begin{cases} Y^{(t+1)} = D^{-1} W Y^{(t)} \\ Y_L^{(t+1)} = Y_L \end{cases}$$

E. LABEL SPREADING

The label spreading (LS) algorithm is similar to LP, except that the labels of $G = \{V, E\}$ may change during the iteration process [51]. Specifically, the clamping factor $\alpha \in (0, 1]$ determines whether the labelled data point will change or not. If $\alpha= 0$, the LS method will not change the original labels, and it will behave as an LP algorithm. The LS algorithm is formulated as follows:

1) Using the kNN method, compute the affinity matrix W.
2) Compute the degree matrix W.

$$D = diag(\left|\sum_j W_{ij}\right| \forall i = 1, 2, .., N + M)$$

3) Compute the normalized graph Laplacian:

$$L = D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$$

4) Choose $\alpha \ \varepsilon(0, 1]$
5) Iterate until convergence is achieve

$$Y^{(t+1)} = \alpha L Y^{(t)} + (1 - \alpha) Y^{(0)}$$

IV. DATA SETS DESCRIPTION

In this study, we considered four publicly available network intrusion datasets to investigate zero-day network intrusion attacks. Each network intrusion dataset contained both benign and malicious network traffic coupled with their respective network traffic features. We used these datasets because they simulated real-world network attacks and clearly labelled features. Furthermore, because the datasets had been used in previous studies, we could leverage the information gained to benchmark our study. Unfortunately, the datasets did not contain zero-day attacks, but we were able to mimic zero-day attacks by splitting a dataset into training (only benign) and testing (only malicious) network traffic [8]–[25]. A dataset may contain various sub-types of data-type features that are either numerical, textual, or binary. Numerical data subtype features have been shown in previous studies to be more effective in differentiating between benign and malicious network traffic, such as zero-day attacks [9]–[29]. Textual and binary data subtypes are static features; therefore, for the datasets used in this study, we only considered the data type of NTA features and, more specifically, numerical subtype features. The UNSW-NB15 network intrusion dataset was generated by the Cyber Range Lab of UNSW Canberra in 2015, using the IXIA PerfectStorm tool [13]. This dataset consists of benign network traffic and nine cyberattacks, which include DoS, generic, worms, shellcode, reconnaissance, fuzzing, analysis, backdoor, and exploit attacks. The network traffic types are described in Table 3 together with the number of instances. In addition, 39 numeric network traffic features (Table 4) were used to describe benign and malicious network traffic. This dataset is publicly available in the CSV format [13].

In Table 4, we considered the numeric features provided by the original authors in [13], and a full description of these features was also provided by the authors. Because some of the features may be redundant, we aimed to identify significant features that could assist in the better detection of zero-day attacks by using Benford's law.

A. CICDDOS2019 DATA SET

The CICDDoS2019 network intrusion dataset was generated by the Canadian Institute for Cybersecurity in 2019 [35]. This dataset contained different types of DDoS and benign networks based on SSH, FTP, HTTP, HTTPS, and email protocols. Each DDoS attack had several benign instances (Table 5). Furthermore, 80 network traffic features (Table 6) were extracted using the CIC flowmeter to describe the benign and DDoS network traffic.

In Table 6, we highlight the original numeric features provided by the authors [35]. Features 5 to 16 are based on packet attributes, features 17 to 22 are based on flow attributes, and so on. We applied Benford's law to identify significant features for differentiating between benign and zero-day network traffic.

B. IOTINTRUSION2020 DATA SET

The IoT network intrusion dataset was developed in [38] in 2020. This dataset contains eight IoT cyberattacks, including flooding, brute force, spoofing, and scanning (Table 7), as well as 79 network traffic features (Table 8) that describe benign and malicious network traffic [38]. Network features were extracted using a CIC flowmeter.

TABLE 11. Summary of significant features identified on the UNSW-NB15 data set by Moustafa *et al.* [13] and Benford's law method.

	Significant features from Moustafa et al. [13] **vs Benford's law**	**Significant / Total features**
Moustafa et al. [13]	2,3,5,7,8,9,10,11,12,16,17,18,20,22,23,24,28,29,30,31,32,33,34,35,37,38,39	27/39
Benford's law	2,3,4,9,10,12,13,16,18,19,24,28,30,33,37,38	16/39

TABLE 12. Summary of significant features identified on the CICDDoS2019 data set by Sharafaldin *et al.* [35] and Benford's law method.

	Significant features from Sharafaldin et al. [35] **vs Benford's law**	**Significant / Total features**
Sharafaldin et al. [35]	2,3,4,7,9,10,12,19,21,22,23,24,26,37,39,41,42,44,50,55,58,66,69,72	24/80
Benford's law	4,8,10,11,12,14,15,19,20,21,22,23,24,26,29,31,41,42,43,44,45,55,67,69	24/80

TABLE 13. Summary of significant features identified on the IoT Intrusion2020 data set by Ullah *et al. [38]* and Benford's law method.

	Significant features from Ullah et al. [38] **vs Benford's law**	**Significant / Total features**
Ullah et al. [38]	1,2,3,9,10,11,13,14,15,33,35,41,42,43,50,54,55,56,57,58,59,60,61,62,63,65,68,69,71	29/79
Benford's law	4,11,12,14,15,17,18,19,20,21,24,39,42,43,44,45,55,56,57,59,63,64,65,66,67,70	26/79

C. CIRA-CIC-DOHBRW-2020 DATA SET

The CIRA-CIC-DoHBrw-2020 dataset was generated by the Canadian Institute for Cybersecurity in 2020 [39]. This dataset contains DoH, non-DoH, benign, and malicious network traffic of the domain name system (DNS) over HTTPS. Benign DoH network traffic instances were generated using Google Chrome and Mozilla Firefox, whereas malicious DoH network traffic instances were generated using iodine, dns2tcp, and DNSCat2 (Table 9). This dataset contains 31 network traffic features (see Table 10).

V. EXPERIMENTAL RESULTS

In this section, we present our feature selection and machine learning model results for all datasets used in this study.

A. FEATURE SELECTION RESULTS

As part of the data cleaning process, we removed negative values (if any) from all the above datasets to apply Benford's law to real positive numbers. By definition, a zero-day attack is an unknown attack; in other words, it is not a part of the training dataset. The common practice of mimicking zero-day attacks is to split a dataset into two groups of known and ''unknown'' (although these may be known) malicious network traffic [8]–[25]. For example, in our experiments, we treated all benign network traffic as known, and all malicious network attacks (i.e. zero-day attacks) as unknown. The aim was then to show that features of benign network traffic closely obeyed Benford's law, while the same features were violated in malicious network traffic. Consequently, such features were considered significant for detecting zero-day attacks. Specifically, we found that a feature is deemed significant if it simultaneously obeys one of Benford's law distributions on benign network traffic and violates Benford's law distributions on malicious network traffic [33], [47], [48]. Network features that did not satisfy this condition were not deemed to be significant, as they failed to differentiate between benign and zero-day network traffic [33], [34] and considered benign and zero-day network traffic to display similar behaviour. Finally, we compared the significant features identified by Benford's law with those identified by the authors for these datasets. In our previous study [40], we benchmarked Benford's law against various feature selection methods to demonstrate its effectiveness, whereas in the current study, we benchmarked only significant features identified by Benford's law against those identified by the authors of each data set.

1) UNSW-NB15 BL REULTS

See Table 11.

2) CICDDOS2019 BL RESULTS

See Table 12.

3) IOTINTRUSION2020 BL RESULTS

See Table 13.

4) CIRA-CIC-DOHBRW-2020 BL RESULTS

See Table 14.

TABLE 14. Summary of significant features identified on the CIRA-CIC-DoHBrw-2020 data set by Montazerishatoori *et al.* [39] and Benford's law method.

	Significant features from Montazerishatoori et al. [39] **vs Benford's law**	**Significant / Total features**
Montazerishatoori et al. [39]	4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31	28/31
Benford's law	3,4,5,6,7,8,9,10,11,12,16,17,18,19,20,24,25,31	18/31

Overall, the significant features identified by Benford's law were fairly consistent with those identified by the authors, as indicated in Table 11, Table 12, Table 13 and Table 14. Moustafa *et al.* [13] identified 27 significant features from a total of 39, whereas Benford's law identified 16 significant features. Of the 27 significant features identified by Moustafa *et al.* [13], 13 overlapped with those identified by Benford's law. Sharafaldin *et al.* [35] identified 24 significant features from 80 features, and Benford's law identified 24 significant features. Of the 24 significant features identified by Sharafaldin *et al.* [35], 14 overlapped with those identified using Benford's law. Ullah *et al.* [38] identified 29 significant features from a total of 79, and Benford's law identified 26 significant features. Of the 29 significant features identified by Ullah *et al.* [38], 11 overlapped with those identified using Benford's law. Lastly, Montazerishatoori *et al.* [39] identified 28 significant features from a total of 31, while Benford's law identified 18 significant features. Of the 31 significant features identified by Montazerishatoori *et al.* [39], 17 overlapped with those identified by Benford's law. Benford's law does not deem the source port, destination port, and protocol features significant for any of the above sets for detecting zero-day attacks. Consequently, we argue that these features are useful for the signature type of NIDS but not for anomaly type ML-NIDS, which are used to detect zero-day attacks. This is because it is practically impossible for an ML-NIDS to know in advance which ports or protocols cybercriminals will use to launch zero-day attacks. A similar finding was discussed in [9]–[29], namely, that features such as protocol type are only useful for detecting specific known attacks and not new zero-day attacks. In addition, Benford's law does not perform well on binary-type features, as Benford's law distributions expect all digits from one to nine to be observable. This explains the difference in the number of significant features identified by Benford's law versus those identified by the respective authors. Examining significant features identified by Benford's law throughout the datasets, features related to network flows, packets, source, and destination features are significant for detecting zero-day attacks. The effectiveness of the selected features is determined by how well an NIDS performs using measures such as precision, recall, F_1 score, and the Matthews correlation coefficient (MCC) [9]. In the next section, we discuss the implementation of various semi-supervised ML models for detecting zero-day attacks and our use of significant features identified by Benford's law method versus the author's significant features for each dataset.

B. MACHINE LEARNING RESULTS

In this section, we describe the implementation of four semi-supervised ML models for detecting zero-day network intrusion attacks. The semi-supervised learning strategy we adopted splits the network dataset into two parts: training and testing datasets [22]–[24]. The training dataset consisted of labelled benign network traffic data, whereas the testing dataset consisted of unlabelled data for both benign and malicious network traffic. We mimicked a zero-day attack by ensuring that the dataset used in the training phase differed from that used in the testing phase [22]–[24]. To summarise, the significant features identified by Benford's law were used as inputs into an ML model, which was subsequently trained using a training dataset to build its knowledge. Subsequently, this learned knowledge was tested on an unseen testing dataset to evaluate its knowledge. We implemented the following semi-supervised ML models: the GMM, OCSVM, and label spreading and propagation. We used a 20–80% train-test split throughout the datasets and used well-known evaluation measures, namely precision, recall, F_1 score, and MCC (derived from Table 15) [52]. The accuracy measure can suffer bias in cases where the dataset is imbalanced [52], as in this study; hence, it was omitted.

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

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

$$F_1 score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

$$MCC\ score = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)\,(TP + FN)(TN + FP)(TN + FN)}}$$

In our experimental setup, we evaluated four semi-supervised ML models to detect network zero-day attacks. For each network intrusion dataset, we evaluated the performance of an ML model using features identified by Benford's law

TABLE 15. Confusion matrix.

	Actual Zero-day	**Actual benign**
Predicted Zero-day	True positive (TP)	False positive (FP)
Predicted benign	False negative (FN)	True negative (TN)

TABLE 16. Detecting Zero-day attacks using the UNSW-NB15 data set based on features in Table 11.

ML model	Precision (%)		Recall (%)		F_1 score (%)		MCC score (%)	
	Benford's law features	Author [13] features	Benford's law features	Author [13] features	Benford's law features	Author [13] features	Benford's law features	Author [13] features
GMM	64	57	97	94	78	71	69	66
OCSVM	84	93	95	65	85	78	74	66
Label spreading	68	64	76	80	63	71	61	62
Label propagation	60	67	78	82	68	74	62	64

TABLE 17. Detecting Zero attacks using the CICDDoS2019 data set based on features in Table 12.

ML model	Precision (%)		Recall (%)		F_1 score (%)		MCC score (%)	
	Benford's law features	Author [35] features	Benford's law features	Author [35] features	Benford's law features	Author [35] features	Benford's law features	Author [35] features
GMM	63	55	98	59	77	51	68	48
OCSVM	75	70	71	73	62	59	61	57
Label spreading	67	64	78	71	62	60	61	59
Label propagation	69	67	83	82	64	74	62	64

TABLE 18. Detecting Zero-day attacks using the IoT Intrusion 2020 data set based on features in Table 13.

ML model	Precision (%)		Recall (%)		F_1 score (%)		MCC score (%)	
	Benford's law features	Author [38] features	Benford's law features	Author [38] features	Benford's law features	Author [38] features	Benford's law features	Author [38] features
GMM	58	55	78	70	59	51	56	50
OCSVM	88	97	80	84	71	78	64	70
Label spreading	60	62	78	85	68	69	61	63
Label propagation	62	60	79	80	68	67	62	64

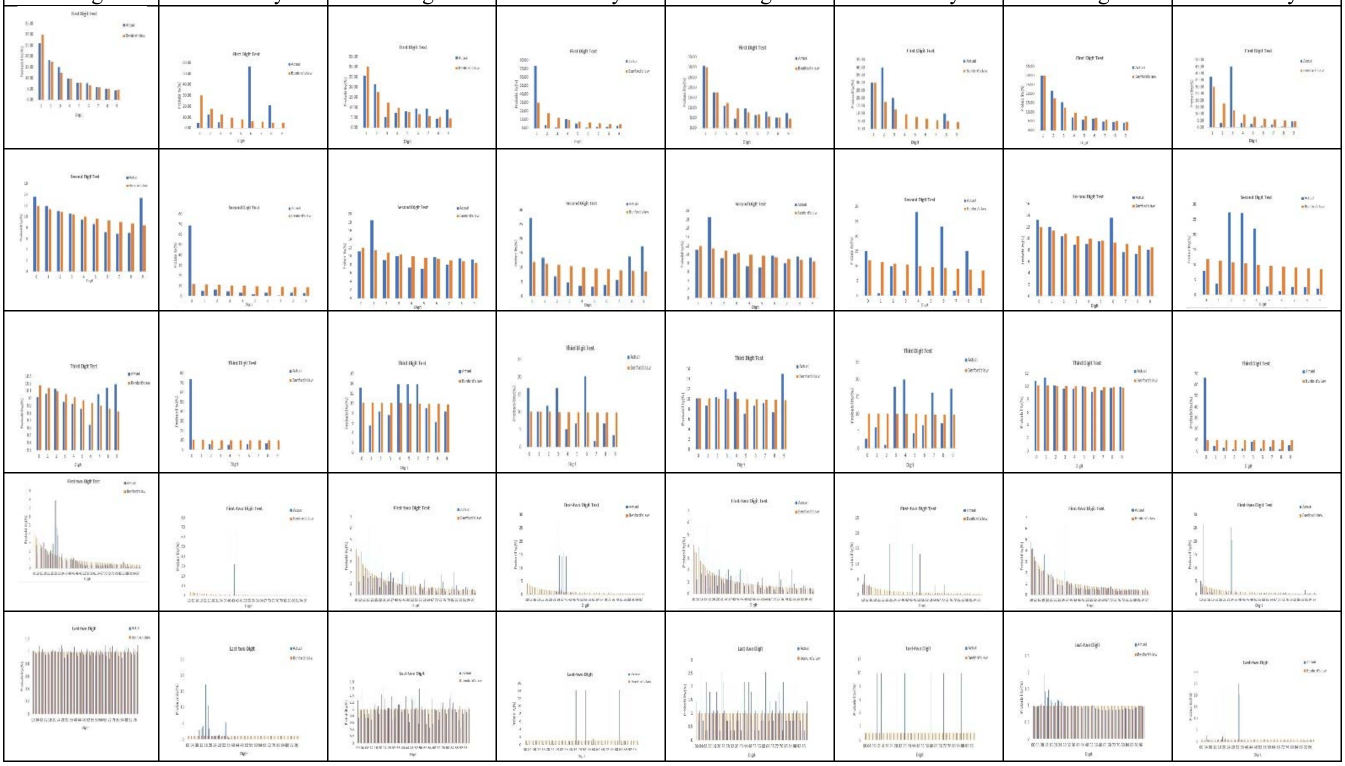
TABLE 19. Detecting Zero-day attacks using the CIRA-CIC-DoHBrw-2020 data set based on features in Table 14.

ML model	Precision (%)		Recall (%)		F_1 score (%)		MCC score (%)	
	Benford's law features	Author [39] features	Benford's law features	Author [39] features	Benford's law features	Author [39] features	Benford's law features	Author [39] features
GMM	61	51	92	82	60	46	58	45
OCSVM	93	94	73	72	78	76	63	62
Label spreading	60	63	78	80	68	71	61	63
Label propagation	66	65	79	75	69	67	63	60

method versus features identified by the respective authors. The results for the UNSW-NB15 dataset are presented in Table 16, those for the CICDDoS2019 dataset in Table 17, for the IoT Intrusion2020 dataset in Table 18 and for the CIRA-CIC-DoHBrw-2020 dataset in Table 19. Overall, the network features identified by Benford's law resulted in better performance than the features identified by the respective authors. This is because features such as the source port, destination port, and protocol are not useful for zero-day detection, and thus negatively impact the performance of an ML model. Moreover, the OCSVM achieved the best results throughout all our experiments, and the Gaussian mixture model performed poorly in terms of precision, recall, F_1 score, and MCC score. Finally, by benchmarking our approach against existing approaches for detecting zero-day attacks (see Table 1), we observed that OCSVM achieved the best MCC of 74%, precision of 84%, recall of 95%, and F_1 score of 85% using the semi-supervised approach. A comparison of our results with those obtained by [27] in Table 1 reveals that our approach yields a slightly better performance. Our

TABLE 20. Benford's law test results of benign and malicious (Zero-day) network attacks.

UNSW-NB15		CICDDoS2019		IoT Intrusion2020		CIRA-CIC-DoHBrw-2020	
Benign	Zero-day	Benign	Zero-day	Benign	Zero-day	Benign	Zero-day



approach can also be compared to DL approaches, such as those adopted in [30], which achieved an F_1 score of 85%. In conclusion, these results demonstrate that semi-supervised approaches are effective in detecting zero-day attacks when features indicative of anomalous behaviour are used. More specifically, the data type features of the numerical subtype are significant for detecting zero-day attacks. Examples of these features include the destination-to-source packet count, source-to-destination bytes, flow duration, packet length, and flow inter-arrival time (IAT).

VI. CONCLUSION

One of the ultimate goals of cybersecurity is to accurately detect zero-day network intrusion attacks in real-time. We first explained how zero-day (unknown) attacks differ from known malicious network traffic and referred to different types of ML approaches that can be employed to detect zero-day attacks. We then focus on the problem of identifying significant features that can differentiate meaningfully between benign and zero-day network traffic. In our study, we propose that the law of anomalous numbers, known as Benford's law, can be used to identify such features. Specifically, this study demonstrated that by analysing the first, second, third, first two, and last two-digit tests of Benford's law, significant features that can discriminate between benign and zero-day network traffic can be identified. We considered four well-known network intrusion datasets covering various types of malicious network traffic. In these datasets, we treated normal network traffic as benign, and malicious network traffic as zero-day attacks. In our experiments, Benford's law was able to identify significant features that distinguished benign and zero-day network traffic. Overall, the significant features identified by Benford's law for detecting zero-day attacks overlapped with the features used to detect known malicious network traffic. However, we observed cases where features such as source ports are important for the detection of known malicious network traffic, but are not deemed significant for the detection of zero-day attacks. This information is not available upfront. In summary, features based on network flow, packets, source, and destination attributes proved to be significant for detecting zero-day attacks. Benford's law distributions are straightforward to implement; therefore, this method is expected to reduce the computational time required to pre-process a high-dimensional imbalanced network traffic dataset. The main limitation of Benford's law is that it is only applicable to positive real numbers, where all digits from zero to nine are observable.

Furthermore, we implemented various semi-supervised ML approaches to detect zero-day network intrusion attacks based on features identified by Benford's law. The experimental results demonstrated that one-class support vector

machines achieved the best results for detecting zero-day attacks with an MCC of 74% and an F1 score of 85%. It became clear from our research that different feature selection methods and ML classifiers can yield different outcomes even on the same network intrusion dataset. The proposed system is applicable to an NIDS. Further research should be conducted to develop a system that combines multiple feature selection methods and ML classifiers. In this way, the best-performing methods are aggregated rather than relying on a single method.

A. APPENDIX – BENFORD's LAW

Examples of FDT, SDT, TDT, F2DT and L2DT.

FDT

Using Equation (1), we can directly compute the probabilities for d ∈ {1,2,..9}.

$$P(D = 1) = \log_{10}\left(1 + \frac{1}{1}\right) = 0.3010$$

SDT

From equation (2), we derive the following equation:

$$\sum_{k=1}^{9} \log_{10}(1+\frac{1}{10k+d}) \text{ where } d = 0, 1, 2, \ldots, 9$$

$$\sum_{k=1}^{9} \log_{10}(1+\frac{1}{10k+0}) = 0.1197$$

TDT

From equation (2), we also derive the following equation:

$$\sum_{k=10}^{99} \log_{10}(1+\frac{1}{10k+d}) \text{ where } d = 0, 1, 2, \ldots, 9$$

$$\sum_{k=10}^{99} \log_{10}(1+\frac{1}{10k+0}) = 0.1018$$

F2DT

We can extend Equation (1) to compute the probability of the first two digits:

$$P(D = 10) = \log_{10}\left(1 + \frac{1}{10}\right) = 0.04139$$

$$P(D = 99) = \log_{10}\left(1 + \frac{1}{99}\right) = 0.00436$$

L2DT

Equation (2) shows that the probability of the last two digits is evenly distributed at approximately 1% [44].

Note that the orange bar lines in Table 20 are Benford's law distributions, and the blue bar lines are the actual distributions. Benign network features closely obey Benford's law, whereas malicious (zero-day) network traffic significantly deviates from Benford's law.

REFERENCES

[1] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Comput. Secur.*, vol. 86, pp. 147–167, Oct. 2019, doi: 10.1016/j.cose.2019.06.005.

[2] M. H. Miraz, M. Ali, P. S. Excell, and R. Picking, "A review on Internet of Things (IoT), internet of everything (IoE) and internet of nano things (IoNT)," in *Proc. Internet Technol. Appl. (ITA)*, Sep. 2015, pp. 219–224.

[3] I. H. Sarker, A. S. M. Kayes, S. Badsha, H. Alqahtani, P. Watters, and A. Ng, "Cybersecurity data science: An overview from machine learning perspective," *J. Big Data*, vol. 7, no. 1, p. 41, Jul. 2020, doi: 10.1186/s40537-020-00318-5.

[4] Y. Diogenes and E. Ozkaya, *Cybersecurity Attack and Defense Strategies: Infrastructure Security With Red Team and Blue Team Tactics*. Birmingham, U.K.: Packt Publishing Ltd, 2018.

[5] A. Thakkar and R. Lohiya, "A review of the advancement in intrusion detection datasets," *Proc. Comput. Sci.*, vol. 167, pp. 636–645, Jan. 2020.

[6] C. Chapman, *Network Performance and Security: Testing and Analyzing Using Open Source and Low-Cost Tools*. Waltham, MA, USA: Syngress, 2016.

[7] A. Yulianto, P. Sukarno, and N. A. Suwastika, "Improving AdaBoost-based intrusion detection system (IDS) performance on CIC IDS 2017 dataset," *J. Phys., Conf. Ser.*, vol. 1192, Mar. 2019, Art. no. 012018, doi: 10.1088/1742-6596/1192/1/012018.

[8] F. Abri, S. Siami-Namini, M. A. Khanghah, F. M. Soltani, and A. S. Namin, "Can machine/deep learning classifiers detect zero-day malware with high accuracy?" in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 3252–3259, doi: 10.1109/BigData47090.2019.9006514.

[9] T. Zoppi, A. Ceccarelli, and A. Bondavalli, "Unsupervised algorithms to detect zero-day attacks: Strategy and application," *IEEE Access*, vol. 9, pp. 90603–90615, 2021, doi: 10.1109/access.2021.3090957.

[10] N. S. Arunraj, R. Hable, M. Fernandes, K. Leidl, and M. Heigl, "Comparison of supervised, semi-supervised and unsupervised learning methods in network intrusion detection system (NIDS) application," *Anwendungen Konzepte Wirtsch.*, vol. 6, pp. 10–19, Jan. 2017.

[11] S. Mukkamala and A. Sung, "Identifying significant features for network forensic analysis using artificial intelligent techniques," *Int. J. Digit. Evidence*, vol. 1, pp. 1–17, May 2003.

[12] K. A. Taher, B. M. Y. Jisan, and M. M. Rahman, "Network intrusion detection using supervised machine learning technique with feature selection," in *Proc. Int. Conf. Robotics, Elect. Signal Process. Techn. (ICREST)*, Jan. 2019, pp. 643–646, doi: 10.1109/ICREST.2019.8644161.

[13] N. Moustafa and J. Slay, "The significant features of the UNSW-NB15 and the KDD99 data sets for network intrusion detection systems," in *Proc. 4th Int. Workshop Build. Anal. Datasets Gather. Exp. Returns Secur. Badgers*, Nov. 2017, pp. 25–31, doi: doi: 10.1109/BADGERS.2015.014.

[14] R. Abdulhammed, H. Musafer, A. Alessa, M. Faezipour, and A. Abuzneid, "Features dimensionality reduction approaches for machine learning based network intrusion detection," *Electronics*, vol. 8, no. 3, p. 322, Mar. 2019, doi: 10.3390/electronics8030322.

[15] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016.

[16] D. Kurniabudi, D. Stiawan, M. Y. Bin Idris, A. M. Bamhdi, and R. Budiarto, "CICIDS-2017 dataset feature analysis with information gain for anomaly detection," *IEEE Access*, vol. 8, pp. 132911–132921, 2020, doi: 10.1109/ACCESS.2020.3009843.

[17] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," *J. Comput. Sci.*, vol. 25, pp. 152–160, Mar. 2017.

[18] P. V. Amoli, T. Hamalainen, G. David, M. Zolotukhin, and M. Mirzamohammad, "Unsupervised network intrusion detection systems for zero-day fast-spreading attacks and botnets," *JDCTA Int. J. Digit. Content Technol. Appl.*, vol. 10, no. 2, pp. 1–13, 2016.

[19] L. Sun, A. Ho, Z. Xia, J. Chen, and M. Zhang, "Development of an early warning system for network intrusion detection using Benford's law features," in *Security and Privacy in Social Networks and Big Data*. Singapore: Springer, 2019, pp. 57–73, doi: 10.1007/978-981-15-0758-8_5.

[20] R. Samrin and D. Vasumathi, "Review on anomaly based network intrusion detection system," in *Proc. Int. Conf. Electr., Electron., Commun., Comput., Optim. Techn. (ICEECCOT)*, Dec. 2017, pp. 141–147.

[21] H. Hindy, D. Brosset, E. Bayne, A. Seeam, C. Tachtatzis, R. Atkinson, and X. Bellekens, "A taxonomy of network threats and the effect of current datasets on intrusion detection systems," *IEEE Access*, vol. 8, pp. 104650–104675, 2020, doi: 10.1109/ACCESS.2020.3000179.

[22] S. Zavrak and M. Iskefiyeli, "Anomaly-based intrusion detection from network flow features using variational autoencoder," *IEEE Access*, vol. 8, pp. 108346–108358, 2020, doi: 10.1109/ACCESS.2020.3001350.

[23] R. Panigrahi and S. Borah, "A detailed analysis of CICIDS2017 dataset for designing intrusion detection systems," *Int. J. Eng. Technol.*, vol. 7, pp. 479–482, Dec. 2018.
[24] H. Hindy, R. Atkinson, C. Tachtatzis, J. N. Colin, E. Bayne, and X. Bellekens, "Utilising deep learning techniques for effective zero-day attack detection," *Electron. Switz.*, vol. 9, no. 10, pp. 1–16, 2020, doi: 10.3390/electronics9101684.
[25] A. Blaise, M. Bouet, V. Conan, and S. Secci, "Detection of zero-day attacks: An unsupervised port-based approach," *Comput. Netw.*, vol. 180, Oct. 2020, Art. no. 107391, doi: 10.1016/j.comnet.2020.107391.
[26] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.
[27] N. H. Duong and H. D. Hai, "A semi-supervised model for network traffic anomaly detection," in *Proc. 17th Int. Conf. Adv. Commun. Technol. (ICACT)*, Jul. 2015, pp. 4–9.
[28] G. Pu, L. Wang, J. Shen, and F. Dong, "A hybrid unsupervised clustering-based anomaly detection method," *Tsinghua Sci. Technol.*, vol. 26, no. 2, pp. 146–153, Apr. 2021, doi: 10.26599/TST.2019.9010051.
[29] Z. Chiba, N. Abghour, K. Moussaid, A. El Omri, and M. Rida, "A novel architecture combined with optimal parameters for back propagation neural networks applied to anomaly network intrusion detection," *Comput. Secur.*, vol. 75, pp. 36–58, Jun. 2018, doi: 10.1016/j.cose.2018.01.023.
[30] N. Abdalgawad, A. Sajun, Y. Kaddoura, I. A. Zualkernan, and F. Aloul, "Generative deep learning to detect cyberattacks for the IoT-23 dataset," *IEEE Access*, vol. 10, pp. 6430–6441, 2022, doi: 10.1109/ACCESS.2021.3140015.
[31] M.-J. Zhu and N.-W. Guo, "Abnormal network traffic detection based on semi-supervised machine learning," *DEStech Trans. Eng. Technol. Res.*, vol. 1, pp. 628–635, Dec. 2017, doi: 10.12783/dtetr/ecame2017/18466.
[32] S. C. Pallaprolu, R. Sankineni, M. Thevar, G. Karabatis, and J. Wang, "Zero-day attack identification in streaming data using semantics and spark," in *Proc. IEEE Int. Congr. Big Data (BigData Congr.)*, Jun. 2017, pp. 121–128, doi: 10.1109/BigDataCongress.2017.25.
[33] E. Druică, B. Oancea, and C. Vâlsan, "Benford's law and the limits of digit analysis," *Int. J. Accounting Inf. Syst.*, vol. 31, pp. 75–82, Dec. 2018, doi: 10.1016/j.accinf.2018.09.004.
[34] M. F. Umer, M. Sher, and Y. Bi, "Flow-based intrusion detection: Techniques and challenges," *Comput. Secur.*, vol. 70, pp. 238–254, Sep. 2017, doi: 10.1016/j.cose.2017.05.009.
[35] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *Proc. Int. Carnahan Conf. Secur. Technol. (ICCST)*, Oct. 2019, pp. 1–8, doi: 10.1109/CCST.2019.8888419.
[36] J. K. Jaiswal and R. Samikannu, "Application of random forest algorithm on feature subset selection and classification and regression," in *Proc. World Congr. Comput. Commun. Technol. (WCCCT)*, Feb. 2017, pp. 65–68.
[37] M. A. M. Hasan, M. Nasser, S. Ahmad, and K. I. Molla, "Feature selection for intrusion detection using random forest," *J. Inf. Secur.*, vol. 7, no. 3, pp. 129–140, 2016.
[38] I. Ullah and Q. H. Mahmoud, "A technique for generating a botnet dataset for anomalous activity detection in IoT networks," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, May 2020, pp. 134–140, doi: 10.1109/SMC42975.2020.9283220.
[39] M. MontazeriShatoori, L. Davidson, G. Kaur, and A. Habibi Lashkari, "Detection of DoH tunnels using time-series classification of encrypted traffic," in *Proc. IEEE Int. Conf Dependable, Autonomic Secure Comput., Int. Conf Pervasive Intell. Comput., Int. Conf. Cloud Big Data Comput., Int. Conf. Cyber Sci. Technol. Congr. (DASC/PiCom/CBDCom/CyberSciTech)*, Aug. 2020, pp. 63–70, doi: 10.1109/DASC-PICom-CBDCom-CyberSciTech49142.2020.00026.
[40] I. Mbona and J. H. P. Eloff. (2021). *Evaluating a Semi-Supervised Intrusion Detection Algorithm Through Benford's Law*. Accessed: Mar. 5, 2022. [Online]. Available: https://www.american-cse.org/static/CSCE21%20book%20abstracts.pdfpage
[41] E. B. Aleksandrova, D. S. Lavrova, and A. V. Yarmak, "Benford's law in the detection of DoS attacks on industrial systems," *Autom. Control Comput. Sci.*, vol. 53, no. 8, pp. 954–962, Dec. 2019, doi: 10.3103/S0146411619080030.
[42] L. Arshadi and A. H. Jahangir, "Benford's law behavior of internet traffic," *J. Netw. Comput. Appl.*, vol. 40, pp. 194–205, Apr. 2014, doi: 10.1016/j.jnca.2013.09.007.
[43] S. J. Miller, *Benford's Law: Theory and Applications*. Princeton, NJ, USA: Princeton Univ. Press, 2015.
[44] M. J. Nigrini, *Forensic Analytics: Methods and Techniques for Forensic Accounting Investigations*. Hoboken, NJ, USA: Wiley, 2012, doi: 10.1002/9781118386798.
[45] A. Iorliam, "Natural laws (Benford's law and Zipf's law) for network traffic analysis," in *Cybersecurity in Nigeria*. Cham, Switzerland: Springer, 2019, pp. 3–22, doi: 10.1007/978-3-030-15210-9_2.
[46] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy*, Jan. 2018, pp. 108–116, doi: 10.5220/0006639801080116.
[47] A. Iorliam, S. Tirunagari, A. T. Ho, S. Li, A. Waller, and N. Poh, "'Flow size difference' can make a difference: Detecting malicious TCP network flows based on benford's Law," 2016, *arXiv:1609.04214*.
[48] K. Sethi, R. Kumar, N. Prajapati, and P. Bera, "A lightweight intrusion detection system using Benford's law and network flow size difference," in *Proc. Int. Conf. Commun. Syst. Netw. (COMSNETS)*, Jan. 2020, pp. 1–6, doi: 10.1109/COMSNETS48256.2020.9027422.
[49] T. Amr, *Hands-on Machine Learning With Scikit-Learn and Scientific Python Toolkits*. Sebastopol, CA, USA: OReilly Media, 2019, p. 384.
[50] P. Perera and V. M. Patel, "Learning deep features for one-class classification," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5450–5463, Nov. 2019, doi: 10.1109/TIP.2019.2917862.
[51] G. Bonaccorso, *Mastering Machine Learning Algorithms*. Birmingham, U.K.: Packt, 2018.
[52] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, no. 1, pp. 1–13, 2020.

INNOCENT MBONA received the B.Sc. degree in mathematics and statistics, the B.Sc. degree (Hons.) in financial mathematics, and the M.Sc. degree in financial engineering from the University of Pretoria, South Africa, where he is currently pursuing the Ph.D. degree in information technology. His research interest includes the use of machine learning techniques to detect cybersecurity threats in computer networks.

JAN H. P. ELOFF received the Ph.D. degree in computer science from Rand Afrikaans University (RAU), South Africa. He is currently the Deputy Dean for research and postgraduate studies at the Faculty of Engineering, Built Environment and IT, University of Pretoria, and also a Full Professor in computer science. He is an internationally recognized Researcher and has published in excess of 113 peer-reviewed papers with a citation count of more than 6000. From 2007 to 2019, he served as an Associate Editor of Elsevier's *Computers & Security* journal.