Journal of Information Technology Management

Print ISSN: 2008-5893

Online ISSN: 2423-5059

Flash Attack Prognosis by Ensemble Supervised Learning for IoT Networks

M. Jagadeesh Babu *

*Corresponding Author, Department of Electronics and Communication Engineering, Jawaharlal Nehru Technological University, Anantapur, Andhra Pradesh, India. E-mail: jagadeeshm.me@gmail.com

A.R Reddy

Retired Senior Scientist- R2, ITI Limited, Bangalore, Karnataka, India. E-mail: ar_reddy@yahoo.com

Abstract

The scope of the Internet of Things (IoT) becomes inevitable in the communication and information-sharing routines of human life, similar to any technological architecture. The IoT is also not exempted from vulnerability to security issues and is even more vulnerable as the networks of IoT are built of non-smart devices. Though the few contributions endeavored to defend against the botnet's attacks on IoT, they partially or poorly performed to defend against the flash crowd or attacks by botnets on IoT networks. In this context, the method “Flash Attack Prognosis by Ensemble Supervised Learning for IoT Networks” derived in this manuscript is centric on defending the flash attacks by botnets. Unlike contemporary models, the proposed method uses the fusion of traditional network features and temporal features as input to train the classifiers. Also, the curse of dimensionality in the training corpus, which is often, appears in the corpus of flash attack transactions by a botnet, has addressed by the ensemble classification strategy. The comparative analysis of the statistics obtained from the experimental study has displayed the significance and robustness of the proposed model compared to contemporary models.

Keywords: Unlike contemporary models; IoT network; Uniform manifold; Classifier.

Journal of Information Technology Management, 2023, Vol. 15, Special Issue, pp. 124-149
Published by University of Tehran, Faculty of Management
doi: https://doi.org/ 10.22059/jitm.2023.91572
Article Type: Research Paper

Received: December 22, 2022
Received in revised form: January 15, 2023
Accepted: February 20, 2023
Published online: March 15, 2023

Introduction

Digital technology, intelligent devices linked to the internet, is heavily used today. Data, threats, and vulnerabilities all grew exponentially as a result. The work produces several large-scale business solutions that depend on open channels or the internet (Brackney, 1998). These criteria support a more productive network environment for end-user e-commerce services. These networks were open to intrusions. As a result, dealing with malicious activities across a wide surface area necessitates network security in the modern world. So, robust security tools are required. Strong IDS are built using machine learning (ML) and artificial intelligence (AI) techniques (Satheesh et al., 2020; Kumar et al., 2020; Rudra Kumar et al., 2022). Attacks by intruders on contemporary solutions based on behavioral analysis or rules are simple. Systems can be trained to recognize attacks using IDS based on ML.

In recent years, ML-based techniques and approaches for data analysis from intelligent applications such as transportation, healthcare, and others have gained popularity. Here, several smart devices produce a large amount of open-channel data. Attackers can now access the internet. Establishing new regulations and rules for mitigating a particular attack type was challenging due to variations in attack vector definitions. It might be necessary to do this when creating new tools and tactics to defend against various attack types.

The daily attack definition variation is one of the most significant issues with IoT. Robust tools are needed to defend against surface attacks. Regular updates to these tools are necessary due to novel attack vectors. Various IDS/tools may pick up on various attacks. One needs to understand the current IDS and its internal architecture to make these devices more efficient. On ML, an algorithm that makes use of training datasets, most IDS were built.

Literature Review

Cyberattacks are on the rise as a result of the IoT connection's increased attention. IDS must examine the network flow of these attacks. Because they can categorize fresh attacks, anomaly detection schemes are crucial. The works (Lunt & Jagannathan, 1988; Tertychny et al., 2020) projected a prototype using abnormal real-time behavior to identify login, connection, IO, CPU, and location & protection damages. IDS requires recognizing network intrusions Axelsson, (2000). An IDS based on anomaly & signature detects unusual behavior by monitoring real-time network activity (Thamaraimanalan, 2021; Kumar et al., 2021). Network-based IDS evasion and mitigation techniques were introduced (Handley et al., 2001). IDS based on signatures can be avoided by polymorphic attach schemes. Due to polymorphic models that might not make an attack normal, IDS based on anomaly offers several protection schemes (Fogla et al., 2006).

IoT-based systems require a quick, secure interface from the internet to embedded devices. Intrusion detection has received more attention due to the dangers and weaknesses in

IoT networks. New intrusion detection methods are needed to address these vulnerabilities. To find IoT anomalies, (Ullah & Mahmoud, 2019) proposed a 2-level hybrid method. The second-level dataset is cleaned using their method, which also employs oversampling, ENN (edited nearer neighbors), and flow-based RFE & Level-1 feature selection. For detecting malicious IoT network activity, their method provided a robust architecture. Modern attackers employ sophisticated tools to carry out dangerous attacks with little expertise. Smart-grid intrusion detection architecture. (Ugtakhbayar et al., 2020; Ullah & Mahmoud, 2017) demonstrate removing unnecessary and redundant features from the NSL-KDD and ISCX datasets using a filter-based feature selection method. Flow-based intrusion detection strategies and difficulties are covered. They divide models into four categories: general, scenario-based, technique-based, and attack-based models (Hofstede et al., 2018).

By utilizing TCP flow to identify and classify malicious behaviors using Benford's law, the work (Satoh et al., 2015) projected IDS based on flow. According to their analysis, each attack has a distinct pattern that they use to differentiate between the abnormal and regular flow. There are numerous intrusion detection techniques which examine packets and exhaust network resources. To identify suspicious network flows, (Zhang et al., 2009) creates semantic links using contextual data. Their prototype successfully distinguished between known and unidentified attacks. For multistep attacks, semantic links increase detection rates. Although semantic links are static, they can be dynamically updated to account for suspicious network flow when spotting new attacks. Compromise devices are found using IDS. Due to the use of SSH for remote server administration, the term "SSH" is used to identify compromised machines. Using SSH, a malicious party can take control of a compromised machine.

For detecting compromised hosts and SSH dictionary attacks, (Koroniotis et al., 2019) proposed flow-based open-source software. They demonstrated their technique in the lab and saw encouraging outcomes. Dictionary stealthy SSH attacks were identified by (Meidan et al., 2018) using features flow and ML. They tested their method on a network campus and found that accuracy rose with computational complexity. IDS research on a computer anomaly is presented by (Jadidi et al., 2013). Four categories - ML, statistical, classification, and determinate state machines were used to categorize models.

Researchers used traditional network datasets to assess IoT networks' methodology because the IoT had a limited IDS dataset. Researchers used the UNSW-NB15, CICIDS2017, and KDD datasets to test their IoT approach. Because they incorporate the internet, mobile networks, fog computing, and cloud computing, IoT networks don't fit the traditional network dataset. IoT devices have constrained processing power and memory. Balachander et al., (2012) suggested using real and fake networks to create a botnet IoT dataset. The weather station, smart thermostat, and remote garage door tools are all red.

Costa et al., (2015) suggested a method for classifying malicious flow using unsupervised density and various sub-space grouping to detect anomalies in flow. A hardware-based BBNN flow identification engine was proposed in (Duque & Omar, 2015). Their hardware-based approach increased computation speed and detection efficiency. A PCA-based anomaly detection method was suggested by (Kumar & Kumar, 2015).

They experimented with their approach on the MAWI dataset and achieved superior outcomes to other anomaly detectors. KNN with density function can enhance anomaly detection. The best K value was determined by PSO, the bat algorithm, gravitational search, and harmony search. The work (Uwagbole et al., 2017) achieved a false negative, a minimum false positive, and an optimal K-means clustering intrusion detection rate. Attacks such as DoS, scanning, and penetration were classified. Their suggested approach examines attack signatures and behavior. NN and GA were combined by (Babu & Reddy, 2020) to find anomalies. We assessed KDD99 and ISCX2012. ISCX2012 had a 97 per cent detection rate using MLP. The NN & DT are used with the KDD99 dataset's reduced features. Ninety-five per cent of regular traffic and 92 per cent of intrusions were detected.

In the field, an IoT SQL Injection Detection is proposed (Moustafa et al., 2018) they tagged logs. Logs give context for SQL injection. They added 862 SQL keywords to the dictionary and extracted 479,000 high-frequency words from the logs. Later, they eliminated duplicate and missing log entries and used SMOTE to balance the data. N-grams are additionally used to extract and choose features. 98.6% accuracy, 99.7% recall, 98.5 % F-measure, and 97.4% precision were attained by the trained SVM method. Models for attack detection were shown in (Hikal & Elgayar, 2020). The contribution is a distributed approach that uses heuristic scales that fit the constrained environment of IoT devices to defend against intrusions at the device level. The other contribution (Muja & Lowe, 2014) utilized net-flow features to defend against botnet attacks at IoT gateways with maximum accuracy and minimal false alarm. Contemporary models become less valuable if the training corpus contains many values projected to network transaction attributes.

Despite false alarms and increased computational overhead, the empirical results of hybrid/ensemble techniques showed better individual performance overall. These contemporary ensemble models have always been more focused on attack detection and have relied on classifier fusion. The dimensionality of the training corpus still plagues fusion classifier training. All classifiers use the same attributes from the training process under consideration.

Methodology

The significant aim of this model is to attain optimal decision accuracy with minimum false alarms in botnet attack detection over IoT networks. The objective of the existing model is a high network transaction and high-dimensional data representation table of features in the training corpus or group. Therefore, the proposed model clusters the specified training corpus for lessening dimensionality. Moreover, optimal features have been derived for every cluster as a self-governing corpus. In further phases, the classifier has been built for every corpus cluster by utilizing optimum features of the resulting training corpus cluster. The last phase or stage of the method predicts a label to input record that allows the suggested labels by the classifiers built from every training corpus.

Moreover, it determines the suitable label, which indicates whether the specified network transaction is prone to attack or benevolent. The block diagram of FAPESL is shown in Figure 1. The descriptions of the below-used formulas are represented in Table 1.

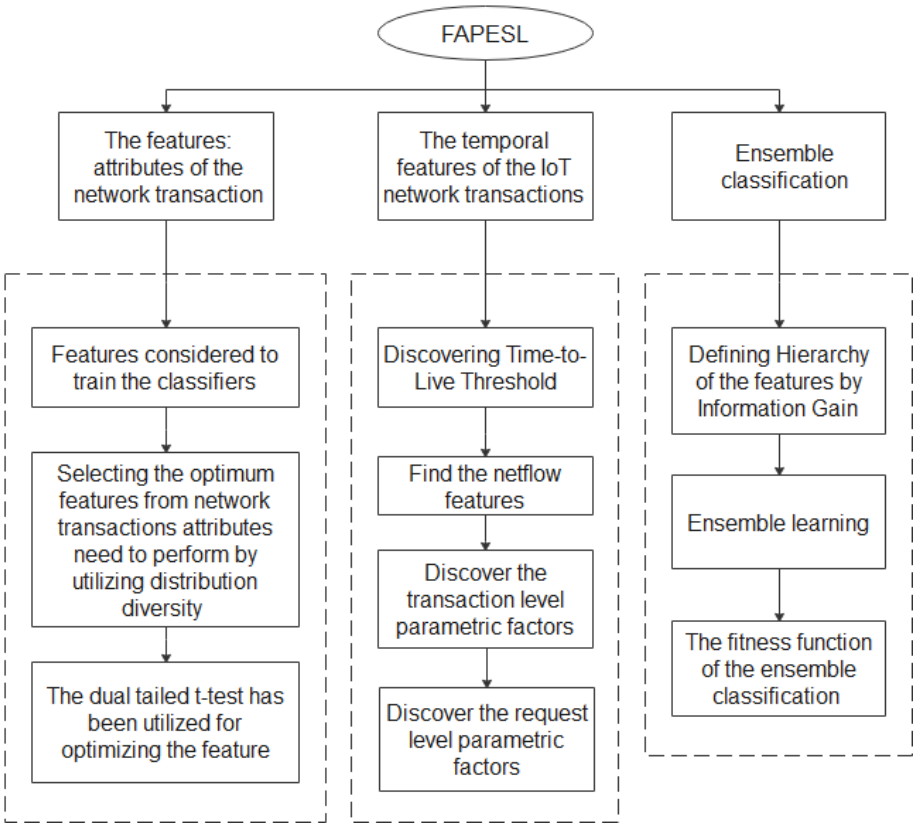

Figure 1. The block diagram representation of FAPESL

Table 1. The descriptions of the formulas

Formula	Description
P	Probabilistic equivalence of high-dimensional data-points
Q	Probabilistic equivalence of low-dimensional data-points
M_{v1}, M_{v2}	Mean values in vectors indicated to be $v1, v2$
C^{i}_{+ve}, C^{j}_{-ve}	Positive and negative label record clusters
$tf(g_i)$	Time-to-Live of the group g
$ct(g_i)$	Completion time
$bt(g_i)$	Begun time
$et(tf_j)$	The end time of the Time-to-Live tf_j
p	Pattern
$IG(p)$	Information gain of pattern p

Uniform Manifold Approximation and Projection

The multiple learning models, uniform manifold approximation and projection (UMAP), aim to present the local structures accurately and include the global structure optimally (Budak & Taşabat, 2016). With the use of massive datasets, UMAP is measured based on three hypotheses called a) data has been distributed uniformly on manifold Riemannian, b) this metric Riemannian is stable locally, and (c) connecting the manifold locally. These predictions probably depict the manifold with a fuzzy topological framework of maximum dimensional data points. While searching for the fuzzy topological framework of low-dimensional data, the embedding manifold has been identified. UMAP depicts data points through a high-dimensional graph for constructing a fuzzy topological framework. UMAP has utilized the exponential probability distribution for computing the equivalence among high-dimensional data points.

$$p_{i|j} = \exp\left(-\frac{d(x_i, x_j) - p_i}{\sigma_i}\right) \quad (1)$$

Here in Equation (1), the representation$d(x_i, x_j)$indicates the distance among i^{th}&j^{th}data points. The notation ρin the above Equation signifies distance amongi^{th} data points and their initial adjacent neighbor. In some instances, the graph weight among i&j nodes are not equivalent to the weight among i&j nodes. The UMAP utilizes high-dimensional possibility, as shown in Equation (2).

$$p_{ij} = p_{i|j} + p_{j|i} - p_{i|j}p_{j|i} \quad (2)$$

Since the construed graph is a likelihood graph, the UMAP requires giving k, which is the representation of nearest or adjacent neighbors in Equation (3).

$$k = 2^{\sum_i p_{ij}} \quad (3)$$

After constructing a high-dimensional graph, the UMAP builds and optimizes the outline of low-dimensional equivalence as much as possible. Here, for modelling distance in the low dimensions, the UMAP utilizes a possibility measure identical to student t-distribution.

$$q_{ij} = \left(1 + a\left(y_i - y_j\right)^{2b}\right)^{-1} \tag{4}$$

In Equation (4), a is equivalent to 1.93, and b is equivalent to 0.79 for UMAP in default.

The UMAP utilizes binary- CE (cross-entropy) in the form of cost-function because of its ability to capture the global data framework.

$$CE(P, Q) = \sum_i \sum_j \left[p_{ij} \log\left(\frac{p_{ij}}{q_{ij}}\right) + \left(1 - p_{ij}\right) \log\left(\frac{1-p_{ij}}{1-q_{ij}}\right)\right] \tag{5}$$

In Equation (5), the notation P indicates the probabilistic equivalence of high-dimensional data points. The notation Q signifies low-dimensional data points.

The cross-entropy derivative has been utilized to update low-dimensional data points coordination for optimizing projection space until convergence. Moreover, UMAP implemented stochastic gradient descent (SGD) because of its rapid convergence, and it lessens the consumption of memory as we calculate the subset gradients of the dataset.

UMAP has several significant hyper-parameters which impact its performance. The hyper-parameters or factors are:

- Target embedding dimensionality.
- The notation k is the number of neighbors, selecting a small value, which indicates that interpretation would be local and a delicate detail framework has been captured. Selecting a considerable value indicates that the prediction would be based on large regions. Therefore, some of the fine-detail frameworks would be missing.
- There is less distance allowed among embedding space points. Minimum values of this less distance would accurately capture the correct manifold structure; however, it might result in dense clouds, making visualization intricate.

Handling the High Dimensionality

The main aim of the projected model of this contribution is to handle the curse of dimensionality in a specified corpus. Concerning this, a clustering algorithm called nearest neighbor graph technology had been adopted for performing clustering, offering a dynamic number of clusters. Moreover, this graph model designs a graph structure where points are the vertices and edges connected to their adjacent neighbors. Here, query points have been

utilized for discussing this graph by utilizing Euclidian distance to be more approximate to adjacent neighbours. Moreover, the corresponding clusters have been optimized by utilizing UMAP because of its achievement in storing global and local structures.

Diversity Assessment by Composite Variance

Here, ANOVA standard T-test has been adapted for evaluating composite variance, which signifies the divergence of features about features spanned amid records of both positive & Negative labels. The t-test is an optimum selection for evaluating composite variance, signifying whether values of 2 distinct sets of the same distribution are varied or identical.

Optimal utilization of the t-test for examining diversity among features in positive and negative label records (Matsuki et al., 2016) is in Equation (6):

$$t - \boldsymbol{score} = \frac{(\boldsymbol{M_{v1} - M_{v2}})}{\sqrt{\frac{\sum_{\boldsymbol{i=1}}^{|\boldsymbol{v1}|}(\boldsymbol{x_i - M_{v1}})^{\boldsymbol{2}}}{\boldsymbol{|v1|-1}} + \frac{\sum_{\boldsymbol{j=1}}^{|\boldsymbol{v2}|}(\boldsymbol{x_j - M_{v2}})^{\boldsymbol{2}}}{\boldsymbol{|v2|-1}}}} \quad (6)$$

- The notation M_{v1}, M_{v2} the above equation depicts mean values in the respective order of vectors indicated to be v_1, v_2. The vectors are made from each dimensional feature as explored in the following:
- For the dimension of the specified feature dim, for every positive label cluster, choose the negative label cluster having similarity with a resulting cluster of a positive label; let full features of the selected dimension identified in records of both negative and positive label clusters in the F_{dim}.
- Moreover, eradicate the replicas from the vector called F_{dim} .
- The possibility of each feature present in the vector F_{dim} have been evaluated about records of positive label cluster and kept in vector v_1 in the same vector sequence F_{dim}.
- Identically, the possibility of each feature present in the vector F_{dim} have been evaluated about records of negative label cluster and kept in vector v_2 in the same vector sequence F_{dim}.
- These vector notations v_1, v_2, the possibility of every feature of a parameter found in both positive and negative label record clusters C_{+ve}^{i}, C_{-ve}^{j}, respectively.

Further, the representations x_i, x_jdepict the values presented in resultant vectors v1 and v2 corresponding to $|v_1|, |v_2|$ sizes.

The test concerns the ratio of resultant vectors' mean variances and the square root of cumulative MSD. Further, the degree of probability indicated the p-value considered for the t-table. Depending on the vectors' features, nature, along with these vectors' features, has been recorded. The less probability p-value exhibits that two vectors were diversified and indicates these vectors feature as optimal.

The Classifier

The random forest algorithm (Moustafa & Slay, 2016) comes under a supervised learning algorithm, generally trained by the bagging model. The indication behind the bagging model is that the integration of learning models enhances the overall outcomes. As the name indicates, a random forest comprises various individual decision trees performing as an ensemble classifier. In a random forest, every tree segregate class prediction and class by majority votes, which becomes our prediction model.

The classifier "Random Forest" is the right choice to learn from the multiple models that are relatively covariant. The model can define as the multiple decision trees built from the relatively covariant models using the randomly picked roots. The other key strength of the Random Forest is that the other trees' classification errors won't influence each tree. The mandate factors required to increase the Random Forest classification process's optimality are portrayed in the following description.

Some signals should be there in our features such that models designed by utilizing those features are better regardless of guessing randomly. Individual trees have made predictions and must have low correlations over each other.

The explanation of random forest for the classification is in the following way:

The algorithm of the random forest for both regression and classification is in the following way:

1. Draw bootstrap samples n_tree from the original data.
2. For every bootstrap sample, increase an unpruned regression tree or classification with the following changes: at every node, instead of selecting the optimal split amid total predictors, random m_try predictors sample and select the optimal split from amid those variables.
3. The novel data need to be predicted by cumulating the predictions or estimations of n_tree trees.

The error rate prediction could be attained based on trained data as follows:

- At a bootstrap iteration, estimate the bootstrap sample's information by utilizing a tree grown in a sample.
- The OOB predictions have to be aggregated. Also, compute the error rate and call it an OOB prediction of the error rate.

The flow of the Random Forest (RF)

Begin RF Algorithm

Input:

The notation N indicates the number of nodes
The notation M signifies the number of features
The notation$D$ indicates the number of trees that have to be built.

The notation Output V indicates the class with the maximum votes
While stopping criteria is false, **do**

Bootstrap sample A should be drawn randomly from training data that is represented with D
The below steps should be used for constructing T_i a tree from a sample Athat is drawn as stated above A:

(1) Choose m features randomly M, whereas $m << M$

(2) Compute the optimal split point amid m features for node d

(3) The node has to be split into two daughter nodes by utilizing an optimal split process

(4) Steps 1, 2, and 3 must be repeated until the required number of nodes has been attained.

Recurring 1-4 steps should build the forest for D times

End While

Output total built trees $\{T_i\}$ $1D$
The novel sample should be applied to every constructed or built tree beginning from the root node.

The sample should be allocated to a class respective to the leaf node.
Associate the overall trees' votes or decisions.
The output is V, which is the highest vote of the class.

End RF Algorithm

Flash Attack Prognosis

Every IoT network transaction transfers complete information regarding source & destination, service protocols, transaction and format, the time needed and time consumed by transaction, the ingress, and egress speed in terms of bytes for one second at source & destination, respectively. Table 2 shows the time-live value of source-destination and vice versa, the transmission of packets between source and destination, along with other features associated with FTP and TCP protocols. Nevertheless, many features were not prominent in determining the transaction fitness towards negative and positive label records. Moreover, the effect of these features lessens slowly for defending botnets. Concerning this, novel features might generate to manage botnet attacks. Hence, the features presented in our one-time contribution

(Muja & Lowe, 2014) have been taken for training this article's projected ensemble classification model. These features have usually been adapted from existing contributions, and the following sections will discuss novel features derived.

The Optimum Features: Optimum Attributes of The Network Transaction

The optimum features from network transaction attributes must be selected by utilizing the distribution diversity of values exhibited for every attribute in both negative and positive labelled records. These are considered for identifying their prominence in the learning procedure of the target ensemble classifier. Here, it suggested temporal features that are both labelled as positive and negative.

Table 2. Optimal features of the network transactions in IoT networks

S.No.	Optimal features in IoT networks
1.	Source to destination transaction bytes
2.	Destination to source transaction bytes
3.	Mean packet size transmitted by a source
4.	Source bits per second
5.	a numeric value derived by the state protocol used and the time to live of the source and destination
6.	Source to destination time to live value
7.	Destination to source time to live value
8.	Total packets per second in transaction
9.	Record total duration of 0.406 d_{mean} Mean packet size transmitted by the destination

The Temporal Features of IoT Network Transactions

This section portrays the temporal features depicted in our earlier contribution BADD, which have been fused with the standard network features.

Discovering the Time-to-Live Threshold

Determining the Time-to-Live Threshold, which represents the lifetime of buffered network transactions, is a crucial step in the proposal's implementation. Records from the training corpus have been used to determine the Time-to-Live cut-off. Each of these files contains information on a transaction that has either a "positive" (vulnerable to attack) or a "negative" (not vulnerable to attack) label (benign transaction). Following is a scaling of the Time-to-Live threshold tft based on the provided training data.

Let's call the set of transactions that have been annotated "positive" (vulnerable to botnet attacks) or "negative" (not vulnerable) C (benign transaction).

As a set C, arrange the transactions mentioned as records in the provided corpus C so that the earliest transaction start time is first. For each specified transaction C, identify all additional transactions whose start times fall before the set transaction's end time.

The following description projects the scheduler's implementation of the stages required for burst buffered clustering in the form of distinct groups.

- Order the time stamps of the transactions you want to list by when they began.
- Select the $b_1$trade, which is most prominently shown as the bcl_1group centroid in the sequencing list.
- Moves in the ordered list towards bcl_1 the group g. Initiation of these transactions occurs at periods that coincide with the duration of the corresponding bcl_1performance time interval.
- After the novel centroid and group bcl_1have been reformed, pick a transaction from the group bcl_1 that has a later end time than the other transactions in the same group. As a result, the transactions in that set have a start time that falls inside the median transmission window.
- A group $bcl_1$is considered final if its members have not changed since it was last locked using the previous group's $g$centroid as a comparison point.
- Repeat the preceding procedures until an ordered list containing the transactions above is impossible to construct.
- A variable number of groups $g$were defined using the provided training data, and the Time-to-Live $tf(g)$ for each group was calculated as the absolute difference between the earliest start time and the latest end time for all transactions in that group.
- Also, compare the observed Time-to-Lives $\langle tf \rangle$ across all of these groups and scale the mean deviation from those values $\langle tf \rangle_d$.

In this case, the recommended time-to-live threshold tft may be calculated as the sum of the means $\langle \mathrm{tf} \rangle$ and standard deviations $\langle \mathrm{tf} \rangle_{\mathrm{d}}$ of the corresponding time-frame tenures.

Assume the transaction clusters in the supplied corpus are represented by the list $\mathrm{G} = \{\mathrm{g}_1, \mathrm{g}_2, \ldots, \mathrm{g}_\mathrm{i}, \mathrm{g}_{\mathrm{i}+1}, \ldots \mathrm{g}_{|\mathrm{G}|}\}$.

Each group's $\{\mathrm{g}_\mathrm{i} \exists \mathrm{g}_\mathrm{i} \in \mathrm{G}\}$ Start

Sort the transactions by the time they began and use the first-indexed transaction's start time $\mathrm{bt(s)}$ as the new baseline.

The Time-to-Live completion time $\mathrm{ct(s)}$ of the first transaction in the list may be determined by sorting the transactions from fastest to slowest.

Calculate the Time-to-Live $tf(g_i)$ for the ggroup by subtracting the End Time $ct(g_i)$ from the Start Time $bt(g_i)$.

End

Determine the Time-to-Live threshold $\{tft \exists tft \equiv \langle tf \rangle + e\}$, which is equal to the average of the observed Time-to-Live $\langle tf \rangle$ across all groups and the observed Time-to-Live eoverall transaction groups, plus their standard deviations.

The NetFlow features

The parametric properties may be uncovered from the set of transactions bound by each Time-to-Live threshold. As a result, the provided training corpus is divided into two groups, C_{+ve}, C_{-ve}, C_{+ve} including transactions that are labelled as positive as well as C_{-ve} negative.

Each group $\{g_i \exists g_i \in G_{+ve}\}or\{g_j \exists g_j \in G_{-ve}\}$in the resulting set G_{+ve}, G_{-ve}comprises transactions with a start time $bt(s)$ that is earlier than the start time $bt(tf_j)$ of the Time-to-Live tf_j and later than the end time $et(tf_j)$ of the Time-to-Live tf_j.

Next, the proposed method finds the values of the parametric features independently for each of the positive label groups $\{g \exists g \in G_{+ve}\}$ and negative label group$\{g \exists g \in G_{-ve}\}$.

Discovering the parametric properties of the positive and negative groups is necessary to expand botnet attacks' scope.

- Source IP confidence: This metric value $\{sIPc(g) \exists sIPc(g) = ucsIP(g) * tcsIP(g)^{-1}\}$denotes the ratio of total distinctive source IP addresses $ucsIP(g)$ in contradiction to the aggregate source IP addresses $tcsIP(g)$
- Source MAC Confidence: This metric value $\{sMc(g) \exists sMc(g) = ucsM(g) * tcsM(g)^{-1}\}$ denotes the ratio of aggregate distinctive source MAC addresses $ucsM(g)$ in contradiction to the aggregate source IP addresses$tcsM(g)$
- Target IP confidence: This metric value $\{tIPc(g) \exists tIPc(g) = uctIP(g) * tcstP(g)^{-1}\}$denotes the ratio of aggregate distinctive target IP addresses $uctIP(g)$ in contradiction to the aggregate target IP addresses$tctIP(g)$

Transaction Level Parametric Factors

- Transaction length minimal confidence is the absolute difference between the ratio of the total transactions to the total transaction length.
- Transaction length confidence (max and min): The max value $\{slc\langle slc(g)\rangle max_{max}\}$ of the group of this metric denotes the sum of the mean of transaction lengths $\langle slc(g)\rangle$ and the respective deviation erroreof the total transactions. Similarly, the min value

$\{\text{slc}|\langle\text{slc(g)}\rangle - \text{e}|\text{min}_{\text{min}}\}$ denotes the absolute difference between the mean of transaction lengths $\langle\text{slc(g)}\rangle$ and the respective deviation error e of the total transactions.

- Request length confidence (max, min): The max $\{\text{rlc}\langle\text{rlc(g)}\rangle\text{max}_{\text{max}}\}$ and $\text{min}\{\text{rlc}|\langle\text{rlc(g)}\rangle - \text{e}|\text{min}_{\text{min}}\}$ values of this metric denote the sum and absolute difference of mean request length$\langle\text{rlc(g)}\rangle$ of the total requests and the respective deviation errore of the request lengths in the corresponding order.
- Transaction level UDP confidence (max, min): The max $\{\text{udpc}\langle\text{udpc(g)}\rangle\text{max}_{\text{max}}\}$and min $\{\text{udpc}|\langle\text{udpc(g)}\rangle - \text{e}|\text{min}_{\text{min}}\}$ values of this metric denote the sum and absolute difference of the mean of the transaction level UDP packets count$\langle\text{udpc(g)}\rangle$ of the total requests and respective deviation error e of the transactions in corresponding order.
- Transaction level TCP confidence (max, min):
- The $\text{max}\{\text{tcpc}\langle\text{tcpc(g)}\rangle\text{max}_{\text{max}}\}$ and min $\{\text{tcpc}|\langle\text{tcpc(g)}\rangle - \text{e}|\text{min}_{\text{min}}\}$ values of this metric denote the sum and absolute difference of the mean of the transaction level TCP packets count $\langle\text{tcpc(g)}\rangle$ of the total requests" and respective deviation error e of the request lengths in corresponding order.
- Transaction level FTP confidence (max, min): The max and $\text{min}\{\text{ftpc}\langle\text{ftpc(g)}\rangle\text{max}_{\text{max}}\}$ $\{\text{ftpc}|\langle\text{ftpc(g)}\rangle - \text{e}|\text{min}_{\text{min}}\}$ values of this metric denote the sum and absolute difference of the mean of the transaction level FTP packets count $\langle\text{ftpc(g)}\rangle$ of the total requests" and respective deviation error e of the transactions in corresponding order.

Request Level Parametric Factors

UDP confidence (max, min), Request level TCP confidence (max, min), and Request level FTP confidence (max, min) shall estimate using the process adopted for transaction-level parametric factors that result following:

$\{udpcr\langle udpcr(g)\rangle max_{max}\}$ $\{udpcr\|\langle udpcr(g)\rangle - e\|min_{min}\}$	The sum and absolute difference of the mean of the request level UDP packets count $\langle udpcr(g)\rangle$ of the total requests and respective deviation error$e$ of the total requests in the corresponding order
$\{tcpcr\langle tcpcr(g)\rangle max_{max}\}$ $\{tcpcr\|\langle tcpcr(g)\rangle - e\|min_{min}\}$	The sum and absolute difference of the mean of the request level TCP packets count $\langle tcpcr(g)\rangle$and the respective deviation error$e$of the total requests in the corresponding order
$\{ftpcr\langle ftpcr(g)\rangle max_{max}\}$ $\{ftpcr\|\langle ftpcr(g)\rangle - e\|min_{min}\}$	The sum and absolute difference of the mean of the transaction level FTP packets count $\langle ftpcr(g)\rangle$ of the total requests and respective deviation error$e$ of the total requests in corresponding order.

Ensemble Classification

The random forest must be constructed using both of these clusters. Each dimension represents a tree, with branching depth proportional to the observed information gained from optimum feature patterns in that dimension. The similarity between the positively labelled clusters C^{i}_{+ve}, C^{j}_{-ve} may be predicted using any similarity metric, including Jaccard's similarity. In addition, the optimal feature patterns for clusters C^{i}_{+ve}, C^{j}_{-ve} are presented below.

To calculate the total number of -1 count subsets, it is necessary to combine the best characteristics from the positive as well as negatively labelled clusters C^{i}_{+ve}, C^{j}_{-ve}into a single set v^{dim}. In this case, the symbol v^{dim} stands for the unique cardinality of the corresponding set. These subsets were identified using a set-indicated vector,v^{dim}. It is used to construct a random forest tree representing clusters C^{i}_{+ve}, C^{j}_{-ve}. Each of the following information gain patterns may be predicted by taking the next step in this scenario.

Finding each pattern's information gain method Cluster $\{p \exists p \in P^{dim}\{\}\}$, connected to C^{i}_{+ve}, C^{j}_{-ve} positive as well as negatively labelled clusters looks like this:

Record clusters C^{i}_{+ve}, C^{j}_{-ve}, which each include records of the positive and negative classes in their order, have ambiguous entropies, as shown by e^{j}_{i}'s entropy. This may be seen in Equation (7).

$$e^{j}_{i} = -\left(\begin{matrix}\frac{|C^{i}_{+ve}|}{|C^{i}_{+ve}|+|C^{j}_{-ve}|}\log_2\left(\frac{|C^{i}_{+ve}|}{|C^{i}_{+ve}|+|C^{j}_{-ve}|}\right)+\\ \frac{|C^{j}_{-ve}|}{|C^{i}_{+ve}|+|C^{j}_{-ve}|}\log_2\left(\frac{|C^{j}_{-ve}|}{|C^{i}_{+ve}|+|C^{j}_{-ve}|}\right)\end{matrix}\right) \quad (7)$$

The entropy of any given $\{p \exists p \in P^{dim}\{\}\}$ the pattern would indeed be zero if the likelihood of pattern pregarding either negative or positive labelled clusters C^{i}_{+ve}, C^{j}_{-ve} is zero. This proves that just one of the possible positive and negative descriptors fits the traditional pattern. If not, then we use Equation (8) to assess alternative entropy patterns indicative of both positive and negative classes.

$$e(p) = -\left(\begin{matrix}\frac{\sum_{m=1}^{|C^{i}_{+e}|}\{1 \exists p \in r_m \wedge r_m \in C^{i}_{+ve}\}}{|C^{i}_{+ve}|}\log_2\frac{\sum_{m=1}^{|C^{i}_{+ve}|}\{1 \exists p \in r_m \wedge r_m \in C^{i}_{+ve}\}}{|C^{i}_{+ve}|}+\\ \frac{\sum_{n=1}^{|C^{j}_{-ve}|}\{1 \exists p \in r_n \wedge r_n \in C^{j}_{-ve}\}}{|C^{j}_{-ve}|}\log_2\frac{\sum_{n=1}^{|C^{j}_{-ve}|}\{1 \exists p \in r_n \wedge r_n \in C^{j}_{-ve}\}}{|C^{j}_{-ve}|}\end{matrix}\right) \quad (8)$$

Additionally, the following evaluates the information-gain IG(p) of pattern p with regards to clusters C^{i}_{+ve}, C^{j}_{-ve}.

$IG(p) = e_i^j - e(p)$	// This expression represents the dissimilarity between the corpus entropy and the dimension dim corresponding to it.

Defining Hierarchy of the Features by Information Gain

- Furthermore, P^{dim} is split into two subsets, $P_o^{dim^{dim}_{\pm ve}}$, P_o^{dim} consisting of patterns in descending order of information gain (thus a zero-entropy pattern).
- All patterns with an entropy greater than zero and a decreasing sequence of information gain may be found in the set $P_{\pm ve}^{dim}$. This is then used to establish the tree's Hierarchy further.
- The initial pattern of information-gain order is from greatest to least, and these levels of the tree hierarchy are maintained separately.
- If the information gained for a given pattern $\{p_i \exists p_i \in P_{\pm ve}^{dim}\{\}\}$ in the positive or negative labelled set $P_{\pm ve}^{dim}$ is more than or equal to the information gained for the given pattern, then the pattern should be maintained at its current hierarchical Hierarchy. Additionally, the patterns are maintained at their present hierarchical level by process of pruning, and this procedure is repeated as necessary to stabilize the remaining patterns' level in the tree's Hierarchy. In addition, the P_o^{dim} patterns in the collection are arranged as if they were the leaves of a tree.
- A node in one level of a hierarchy can be a child node to one or more nodes at a higher level of the Hierarchy depending on the general patterns at that level.
- In addition, the random forest has to be determined for each pair of clusters $C_{+ve}^{i}, C_{-ve}^{j}$by employing an RF classifier that focuses on individual characteristics, as will be elaborated upon below.

Ensemble Learning

It is necessary to build the random forest T for both the positive as well as negative clusters C^{i}_{+ve}, C^{j}_{-ve}. Here, the optimal features with information gain are arranged into distinct hierarchical levels.

The Fitness Function of Ensemble Classification

Preprocessing and feature collection for the given record r are accomplished in the first stage. This is the sum of all the feasible feature subsets in dimension dim, based on the data in the record r.

- Finding fitness at the feature level is done in the following fashion for each multi-branch tree with MFT parallel branches for each dimdimension.
- From the root to each leaf node, there are pc_T^{dim} possible routes.
- Possible number of ways exists for a node at the tree's centre to reach its leaves pc_{+ve}^{dim} is specified by its positive label
- The negative label record pc_{-ve}^{dim} represents the number of paths from the root node to the leaf nodes.
- As demonstrated in Equation (9), the pathways from the root node to the leaf nodes are labelled as positive pr_{+ve}^{dim}.

$$\boldsymbol{pr}_{+ve}^{\boldsymbol{dim}\frac{\boldsymbol{pc}_{+ve}^{\boldsymbol{dim}}}{\boldsymbol{pc}_{T}^{\boldsymbol{dim}}}} \tag{9}$$

- As indicated in Equation (10), the likelihood of the route pr_{-ve}^{dim} from the root node to the negative labeled leaf nodes is as follows.

$$\boldsymbol{pr}_{-ve}^{\boldsymbol{dim}\frac{\boldsymbol{pc}_{-ve}^{\boldsymbol{dim}}}{\boldsymbol{pc}_{T}^{\boldsymbol{dim}}}} \tag{10}$$

- Finds entropy $e_T^{dim}$of the tree T as shown in Equation (11):

$$\boldsymbol{e}_{\boldsymbol{T}}^{\boldsymbol{dim}\left(\boldsymbol{pr}_{+ve}^{\boldsymbol{dim}\log_2\left(\boldsymbol{pr}_{+ve}^{\boldsymbol{dim}}\right)}\boldsymbol{pr}_{-ve}^{\boldsymbol{dim}\log_2\left(\boldsymbol{pr}_{-ve}^{\boldsymbol{dim}}\right)}\right)} \tag{11}$$

- A search should be performed on a branch depicting the specified feature $dim$to discover the potential paths from the root towards the leaf.
- Find the number of paths (denoted as ct_{+ve}) to reach the leaves labeled as positive.
- Identify the path probability pr_{+ve}^{r} to reaching leaf nodes of the positive label as shown in Equation (12).

$$\boldsymbol{pr}_{+ve}^{\boldsymbol{r}} = \frac{\boldsymbol{ct}_{+ve}}{\boldsymbol{pc}_{+ve}^{\boldsymbol{dim}}} \tag{12}$$

- Find out how many ways there are to go to the negatively marked leaf node (hence referred to as ct_{-ve}).
- Use Equation (13) to determine the probability of a route to the negative-labelled leaf nodes (denoted pr_{-ve}^{r}).

$$\boldsymbol{pr}_{-ve}^{\boldsymbol{r}} = \frac{\boldsymbol{ct}_{-ve}}{\boldsymbol{pc}_{-ve}^{\boldsymbol{dim}}} \tag{1}$$

- Using Equation (14), find the entropy (denoted e_r^{dim}) of a particular record r concerning the tree T.

$$e_r^{dim(pr^r_{+ve}\, log_2(pr^r_{+ve})+pr^r_{-ve}\, log_2(pr^r_{-ve}))} \quad (2)$$

- If all counters are zero, then the entropy of the test record r at the same level of the tree$T$ is also zero, indicated by e_r^{dim}.
- Determine the test record r's information gain relative to the Hierarchy dim of a given tree, using Equation (15).

$$IG_T^{dim_T^{dim_r^{dim}}} \quad (3)$$

- If the entropy of the tree's feature hierarchy (T) is equal to the information gain of the test record (IG_T^{dim}), then the following should be done.
- Record fitness at the feature hierarchical level towards the positive label is one, in addition to fitness under the label negative being zero, if there is a path probability (denoted as IG_T^{dim}) between positive labeled leaf nodes and the root node of the corresponding tree concerning the test record r. The fitness of the negative label is one and the fitness of the positive label is zero if and only if the associated route probability (IG_T^{dim}) is zero.
- Alternately, the fitness of the specified record (denoted as r) is evaluated towards positive and negative label records using Equation (16) if the information gain (denoted as IG_T^{dim}) of the record is smaller than the entropy (denoted as e_T^{dim}) of the feature hierarchy dim of the tree$T$.

$$\frac{pr^r_{+ve}}{(pr^r_{+ve}+pr^r_{-ve})}, \frac{pr^r_{-ve}}{(pr^r_{+ve}+pr^r_{-ve})} \quad (4)$$

The absolute variance of the average fitness and MSE of the related label represents the lower bound of the positive or negative label fitness of the test record r at the hierarchical feature level. Furthermore, the record's label is determined by summing its positive and negative fitness across all levels of the hierarchy. According to the total fitness value of the feature hierarchies for both positive and negative labels, a label has been assigned to the associated test record.

Results and Discussion

The tests were conducted using the dataset UNSW-NB15, which is representative of the coexistence of legal user network traffic and attack traffic. 321283 records are vulnerable to attacks, and 2308760 are not. The curse of dimensionality may be shown with the help of Bootstrap Aggregation, which the 321283 reliable records have considered.

Performance Analysis

The metrics utilized to evaluate the proposed technique in comparison to state-of-the-art methods are discussed below, as are the results of research comparing the observed statistics

to the equivalent metrics from the suggested as well as state-of-the-art models described in BADD, EASFF, as well as EDPT.

Measures

Confusion-matrix, which provides results of counting mistakenly and correctly classified occurrences for every event class, is the basis for developing successful classification methods. That's why we use statistical thinking and carefully consider the spectrum of determined measures to make the best possible decisions.

To provide context for performance indicators, the confusion matrix used the following four metrics: True positives (TP) represent the number of accurate optimistic predictions made on a testing set. In contrast, true negatives (TN) represent the number of accurate negative predictions made on a training set. In contrast, false positives (FP) represent the number of inaccurate optimistic estimations made on a testing set.

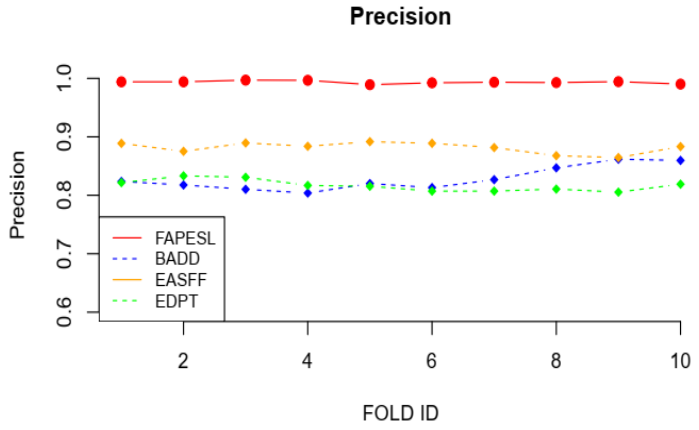

Figure 1. Comparative analysis of precision observed from suggested and existing models

Metric precision and tenfold cross-validation are plotted in Figure 2 over the suggested FAPES model and the current models EDPT, EASFF, and BADD. Precision, also known as a positive predictive value, calculates the ratio of true positives against predicted positives. The suggested FAPESL model is statistically more effective than existing models.

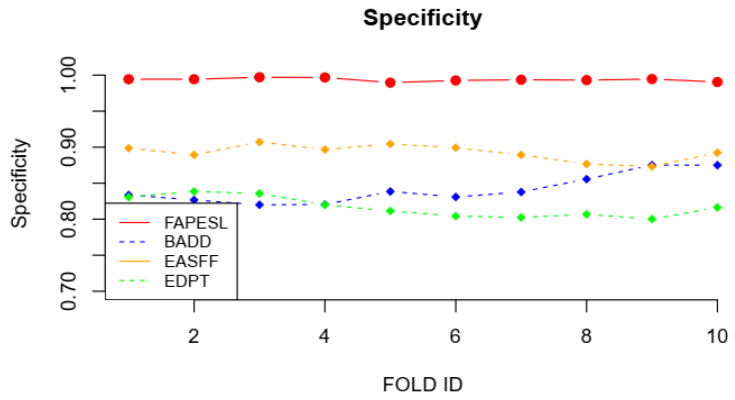

Figure 2. Comparative analysis of specificity observed from suggested and existing models

One of the cross-validation metrics is specificity, measured as the proportion of indeed predicted negatives and the total number of negatives taken into account. Figure 3 illustrates the graph between this specificity measure and tenfold cross-validation of the suggested FAPESL and existing EASFF, EDPT, and BADD. The metric values demonstrate that the suggested model outperforms earlier models.

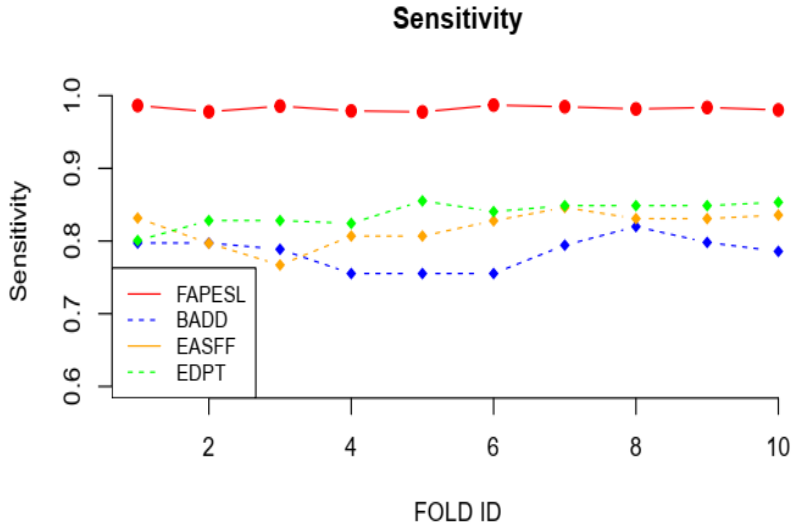

Figure 3. Comparative analysis of sensitivity observed from suggested and existing model

In Figure 4, the suggested FAPES model and existing EDPT, EASFF, and BADD models are compared in terms of metric sensitivity observed from tenfold cross-validation. Sensitivity, often termed recall, denotes the proportion of indeed predicted positives to total positives that were taken into account. It is predicted from statistics that the suggested FAPESL model's sensitivity is superior to that of existing models.

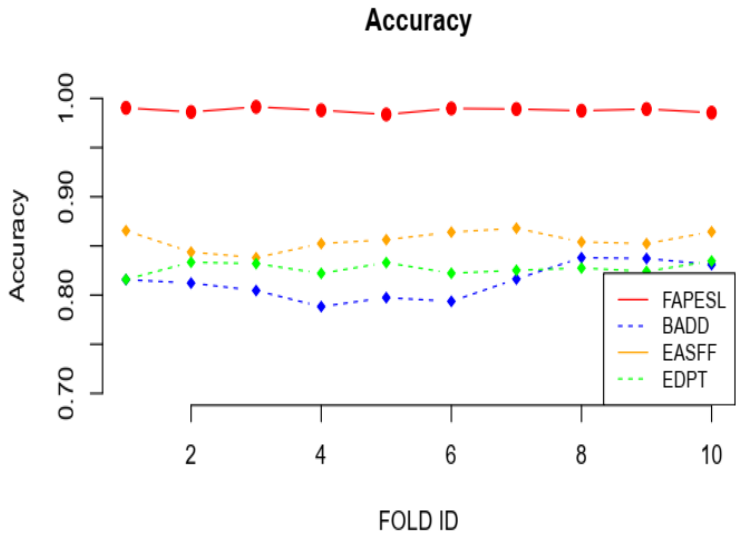

Figure 4. Comparative analysis of accuracy observed from suggested and existing models

One of the cross-validation measures, accuracy, is defined as the proportion of indeed predicted positives and negatives to the total of the true positives and negatives taken into account. Figure 5 illustrates the accuracy observed from tenfold cross-validation of the suggested FAPESL model and existing EASFF, EDPT, and BADD. The analyses showed that, when compared to the existing models, the accuracy of the suggested model is better.

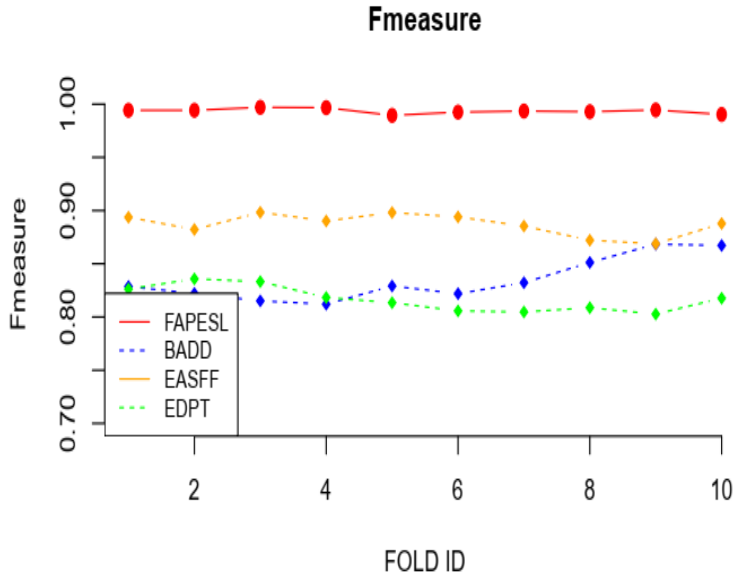

Figure 5. Comparative analysis of F-measure observed from suggested and existing models

The proposed FAPES model and existing models EDPT, EASFF, and BADD on a graph among metric F-measure and tenfold cross-validation in Figure 6. According to statistics, the suggested FAPESL model's F-measure is expected to be more significant than the existing EDPT, BADD, and EASFF models.

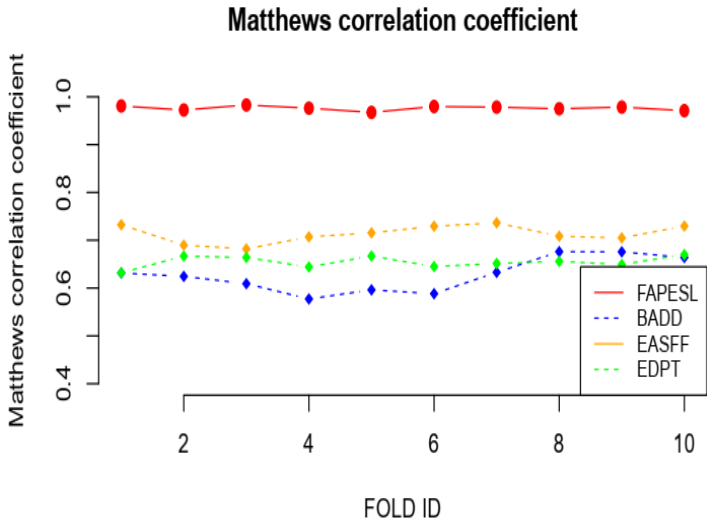

Figure 7. Comparative analysis of MCC observed from suggested and existing models

The MCC metric is one of the cross-validation measures used for binary classification evaluation. According to Figure 7, a graph is drawn between the values for the MCC metric and ten folds of the cross-validation performed on suggested FAPESL and existing EDPT, EASFF, and BADD. The data indicate that the MCC for the suggested model outperformed the existing models.

Table 3. Disparities in efficiency among FAPESL as well as other modern types

a) Distinct differences in T-score and corresponding p-value for precision, as well as specificity among suggested and existing models

Precision			**Specificity**		
	T-score	p-value		T-score	p-value
FAPESL & BADD	24.0766	< .00001	FAPESL & BADD	22.1386	< .00001
FAPESL & EA-SFF	35.1217	< .00001	FAPESL & EA-SFF	28.1875	< .00001
FAPESL & EDPT	60.1325	< .00001	FAPESL & EDPT	41.9026	< .00001

b) Distinct differences in T-score and corresponding p-value for sensitivity as well as accuracy among suggested and existing models

Sensitivity			Accuracy		
	T-score	p-value		T-score	p-value
FAPESL & BADD	28.5736	< .00001	FAPESL & BADD	30.6097	< .00001
FAPESL & EA-SFF	22.4879	< .00001	FAPESL & EA-SFF	40.6491	< .00001
FAPESL & EDPT	24.6972	< .00001	FAPESL & EDPT	64.9021	< .00001

c) Distinct differences in T-score and corresponding p-value for F-Measure as well as MCC among suggested and existing models

F-measure			MCC		
	T-score	p-value		T-score	p-value
FAPESL & BADD	23.2056	< .00001	FAPESL & BADD	30.4196	< .00001

FAPESL & EA-SFF	31.7198	< .00001	FAPESL & EA-SFF	42.7186	< .00001
FAPESL & EDPT	49.9502	< .00001	FAPESL & EDPT	64.3264	< .00001

The t-test applied to the values of performance measures acquired via suggested FAPESL and existing techniques BADD, EASFF, and EDPT provides confidence in the consistency of the predicted method's performance. Table 3 compares FAPESL to other existing models based on the t-score and the accompanying probability value perceived for different metric values. When comparing FAPESL to the other methods, a positive t-score indicates a degree of correlation close to 0, suggesting that FAPESL is more likely than the other methods. To that end, it could seem reasonable to infer that the proposed FAPESL technique outperforms the set of competing approaches carefully considered in the evaluation. When compared to other modern approaches, the EASFF, EDPT, and BADD all perform admirably in their respective categories.

Conclusion

The flash attacks by botnets over IoT have been addressed in this manuscript. The proposed method, “Flash Attack Prognosis by Ensemble Supervised Learning for IoT Networks,” has handled the flash attacks by addressing the crucial objectives. They are listed as adapting temporal features derived from the net flows to train the classifier, handling the curse of dimensionality that often appears in flash crowd network transactions. The ensemble classification has been carried out using Random Forest for each cluster of the training corpus. The experimental study on the proposed model and the other contemporary models portrays the proposed model FAPESL over the other contemporary models. Future research can extend this contribution by ensemble the multiple feature optimization methods to achieve increased accuracy with balanced specificity and sensitivity. The other dimension of future research can use soft computing techniques to improvise the optimal feature selection and prognosis of flash attacks.

Conflict of interest

The authors declare no potential conflict of interest regarding the publication of this work. In addition, the ethical issues including plagiarism, informed consent, misconduct, data fabrication and, or falsification, double publication and, or submission, and redundancy have been completely witnessed by the authors.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article

References

Axelsson, S. (2000). The base-rate fallacy and the difficulty of intrusion detection. ACM Transactions on Information and System Security (TISSEC), 3(3), 186-205.

Babu, M. J., & Reddy, A. R. (2020). SH-IDS: specification heuristics based intrusion detection system for IoT networks. Wireless Personal Communications, 112(3), 2023-2045.

Balachander, K., Venkatesan, C., & Kumar, R. (2021). Safety driven intelligent autonomous vehicle for smart cities using IoT. International Journal of Pervasive Computing and Communications.

Brackney, R. (1998, October). Cyber-intrusion response. In Proceedings Seventeenth IEEE Symposium on Reliable Distributed Systems (Cat. No. 98CB36281) (pp. 413-415). IEEE.

Budak, H., & Taşabat, S. E. (2016). A modified t-score for feature selection. Anadolu University Journal of Science and Technology A-Applied Sciences and Engineering, 17(5), 845-852.

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

Duque, S., & bin Omar, M. N. (2015). Using data mining algorithms for developing a model for intrusion detection system (IDS). Procedia Computer Science, 61, 46-51.

Fogla, P., Sharif, M. I., Perdisci, R., Kolesnikov, O. M., & Lee, W. (2006, August). Polymorphic Blending Attacks. In USENIX security symposium (pp. 241-256).

Handley, M., Paxson, V., & Kreibich, C. (2001). Network Intrusion Detection: Evasion, Traffic Normalization, and {End-to-End} Protocol Semantics. In 10th USENIX Security Symposium (USENIX Security 01).

Hikal, N. A., & Elgayar, M. M. (2020). Enhancing IoT botnets attack detection using machine learning-IDS and ensemble data preprocessing technique. In Internet of Things—Applications and Future (pp. 89-102). Springer, Singapore.

Hofstede, R., Pras, A., Sperotto, A., & Rodosek, G. D. (2018). Flow-based compromise detection: lessons learned. IEEE security & privacy, 16(1), 82-89.

Jadidi, Z., Muthukkumarasamy, V., & Sithirasenan, E. (2013, August). Metaheuristic algorithms-based flow anomaly detector. In 2013 19th Asia-Pacific Conference on Communications (APCC) (pp. 717-722). IEEE.

Kanda, Y., Fontugne, R., Fukuda, K., & Sugawara, T. (2013). ADMIRE: Anomaly detection method using entropy-based PCA with three-step sketches. Computer Communications, 36(5), 575-588.

Koroniotis, N., Moustafa, N., Sitnikova, E., & Turnbull, B. (2019). Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. Future Generation Computer Systems, 100, 779-796.

Kumar, G., & Kumar, K. (2015). A multi-objective genetic algorithm based approach for effective intrusion detection using neural networks. In Intelligent methods for cyber warfare (pp. 173-200). Springer, Cham.

Kumar, M. R., & Gunjan, V. K. (2020). Review of machine learning models for credit scoring analysis. Ingeniería Solidaria, 16(1).

Kumar, M. R., Chakravarthy, V. D., Ranganatham, T. N., & Ramana, K. (2021). Personal finance transaction index scoring using machine learning model. Materials Today: Proceedings.

Lunt, T. F., & Jagannathan, R. (1988, April). A prototype real-time intrusion-detection expert system. In IEEE Symposium on Security and Privacy (Vol. 59, pp. 18-21).

Matsuki, K., Kuperman, V., & Van Dyke, J. A. (2016). The Random Forests statistical technique: An examination of its value for the study of reading. Scientific Studies of Reading, 20(1), 20-33.

Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Shabtai, A., Breitenbacher, D., & Elovici, Y. (2018). N-baiot—network-based detection of iot botnet attacks using deep autoencoders. IEEE Pervasive Computing, 17(3), 12-22.

Moustafa, N., & Slay, J. (2016). 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), 18-31.

Moustafa, N., Turnbull, B., & Choo, K. K. R. (2018). An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things. IEEE Internet of Things Journal, 6(3), 4815-4830.

Muja, M., & Lowe, D. G. (2014). Scalable nearest neighbor algorithms for high dimensional data. IEEE transactions on pattern analysis and machine intelligence, 36(11), 2227-2240.

Rudra Kumar, M., Pathak, R., & Gunjan, V. K. (2022). Machine Learning-Based Project Resource Allocation Fitment Analysis System (ML-PRAFS). In Computational Intelligence in Machine Learning (pp. 1-14). Springer, Singapore.

Satheesh, N., Rathnamma, M. V., Rajeshkumar, G., Sagar, P. V., Dadheech, P., Dogiwal, S. R., ... & Sengan, S. (2020). Flow-based anomaly intrusion detection using machine learning model with software defined networking for OpenFlow network. Microprocessors and Microsystems, 79, 103285.

Satoh, A., Nakamura, Y., & Ikenaga, T. (2015). A flow-based detection method for stealthy dictionary attacks against Secure Shell. Journal of Information Security and Applications, 21, 31-41.

Tertytchny, G., Nicolaou, N., & Michael, M. K. (2020). Classifying network abnormalities into faults and attacks in IoT-based cyber physical systems using machine learning. Microprocessors and Microsystems, 77, 103121.

Thamaraimanalan, T., Mohankumar, M., Dhanasekaran, S., & Anandakumar, H. (2021). Experimental analysis of intelligent vehicle monitoring system using Internet of Things (IoT). EAI Endorsed Transactions on Energy Web, 8(36).

Tran, Q. A., Jiang, F., & Hu, J. (2012, June). A real-time netflow-based intrusion detection system with improved BBNN and high-frequency field programmable gate arrays. In 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (pp. 201-208). IEEE.

Ugtakhbayar, N., Usukhbayar, B., & Baigaltugs, S. (2020). A hybrid model for anomaly-based intrusion detection system. In Advances in Intelligent Information Hiding and Multimedia Signal Processing (pp. 419-431). Springer, Singapore.

Ullah, I., & Mahmoud, Q. H. (2017, December). A filter-based feature selection model for anomaly-based intrusion detection systems. In 2017 IEEE International Conference on Big Data (Big Data) (pp. 2151-2159). IEEE.

Ullah, I., & Mahmoud, Q. H. (2019, January). A two-level hybrid model for anomalous activity detection in IoT networks. In 2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC) (pp. 1-6). IEEE.

Uwagbole, S. O., Buchanan, W. J., & Fan, L. (2017, May). Applied machine learning predictive analytics to SQL injection attack detection and prevention. In 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM) (pp. 1087-1090). IEEE.

Zhang, W., Yang, Q., & Geng, Y. (2009, January). A survey of anomaly detection methods in networks. In 2009 International Symposium on Computer Network and Multimedia Technology (pp. 1-3). IEEE.

Bibliographic information of this paper for citing:

Jagadeesh Babu, M. & Reddy, A.R. (2023). Flash Attack Prognosis by Ensemble Supervised Learning for IoT Networks. *Journal of Information Technology Management*, 15 (Special Issue), 124-149. https://doi.org/ 10.22059/jitm.2023.91572