Robustness of Hybrid IDS Models In Encrypted Traffic Classification

Rhea Sundaresan (19MA20063)
2nd Year Undergraduate Student
Department of Mathematics
IIT Kharagpur

Under the guidance of:
Prof. Arash H. Lashkari,
University of New Brunswick, Canada

July 2021

Contents

1 Introduction

Encrypted traffic classification has become important with the increase in popularity of various traffic encryption methods like Tor and VPN. Two key areas of interest when it comes to machine-learning based Intrusion Detection Systems (IDS) have been minimizing false positive rates (FPR) and building models that are robust to adversarial attacks. In this study, we propose a 2-step hybrid IDS model that can achieve both objectives, by utilising feature-splitting into 2 categories - flow based, and packet based. We use adversarial training as our defence mechanism against attacks. Our IDS model was trained on the CIC-Darknet 2020 Dataset, appropriately re-labelling the Level 1 label to classify encrypted traffic in a binary method.

The contributions of this study are as follows:

1. Proposing a hybrid IDS model for encrypted traffic which greatly reduces FPR while maintaining accuracy, using a flow/packet based feature division.
2. Generating various black-box attack samples measuring the model's performance against the same, in this process showing that it is robust, accurate and precise.
3. Finding meaningful patterns in observing results with the Jacobian Saliency Map Attack samples by tuning its two hyperparameters, to replicate the way in which real-life adversaries would attack.

2 Literature Review

In this section, we examine and summarise past works in the field of traffic classification, as well as more recent studies focusing on adversarial attacks in machine learning.

2.1 Darknet & Traffic Classification

Studying darknet traffic has been important for detection of several cyber attacks and other types of malicious traffic. The reason for this being, Darknet IP addresses do not have any active servers, and hence cannot send out any traffic of their own. Hence, all traffic entering into a Darknet can be said to be large-scale, or illegitimate. Initial studies focused on port-based classification, but issues like port multiplexing and port hopping soon switched the focus to machine-learning based classification instead. Furutani et. al. [1] aimed to classify TCP packets in the darknet into backscatter caused by DDoS attacks or normal traffic. To achieve this, they defined 11 features for TCP packets, based on IP and port, and ran a SVM classifier with RBF kernel, and were able to get detection accuracy greater than 90%, with varying time intervals of observation going from 15 to 90 seconds.

Liu et. al. [2] attempted to extend similar logic, to see the effectiveness of a darknet taxonomy for real long-range traffic. They divided anomalous traffic into 9 categories, on the basis of certain traffic rules. Applying this metric on long-range data i.e, 74 weeks worth of data, they achieved a labelling and detecting rate of greater than 96% for all sources. They were also able to demarcate the most popular TCP ports exploited by attackers.

In this study by Nishikaze et. al. [3], they suggest a machine learning approach to large-scale monitoring for malicious activities on the Internet. They did this by dividing the source IP into subnets, and further transforming the network traffic in each subnet into a feature vector based on the TAP (Traffic Analysis Profile) analysis, that classifies short-time network traffic into 27 categories based on the numbers of packets, destination IPs/ports, and source IP/ports. A hierarchial clustering method was adopted, in which two clusters with the closest distance are merged until the cluster distance reaches a threshold value. Any incoming packets were converted into feature vectors, and the nearest cluster was identify to classify the traffic into an attack type. The study found an interesting transition of TAP feature clusters for some adjacent subnets, which had distinctive transitions from the other subnets. The detailed packet analysis implies that this distinctive transition was related to the pandemic of the attack to the vulnerability of Synology NAS (port 5,000/TCP). This indication of this attack was detected on the proposed monitoring system about 20 days earlier than the pandemic, showing that the system was promising as a large-scale monitoring system for cyber-attacks. Still, they needed to prove that the proposed system could capture indications of other pandemics of unknown malwares.

In the previous study, a limitation was the lack of clarity in suiting a threshold value for the hierarchical clustering phase. In tandem with this, Bou-Harb et. al. [4] proposed a pre-processing tool which was more formal, did not depend on any hard thresholds, did not make any assumptions on the nature of the misconfiguration traffic, and was shown to be more accurate. The model assesses the difference of the probability estimates corresponding to misconfiguration, other non-malicious traffic, and malicious traffic, and outputs its result on the basis of the higher likelihood.

Vichaidis et. al. [5] propose a methodology which involves categorizing the monitored traffic into multiple components based on various criteria, including IP address, port number, and other fields in the header of each packet. They introduce the concept of *TopN* values, and after setting an appropriate threshold limit, show that sudden changes in *TopN* are an important way of identifying attacks using traffic stability.

Han et. al. [6] proposed an online processing algorithm of the *GLASSO* engine and undertook outlier detection sequentially. It was possible to detect outliers with a very short span compared to earlier work and obtain alert results in real time. The engine produced alerts for three types of activities, i.e., cyberattacks, survey scans, and sporadically-focused traffic, and that they can be properly distinguished by preparing the proper criteria. By analyzing the alerts obtained from the *GLASSO* engine for one month in October 2018 for each TCP port, 128 TCP ports were obtained among the total 1,634 alerts, further classified into the 3 aforementioned classes.

An interesting piece of research by Gadhia et. al. [7], examines incoming traffic on two darknet sensors in different locations. The authors collected packet data across 91 days, and analysed source-IP and destination port data. The authors found that TCP traffic remains almost comparable in both the sensors, whereas UDP traffic varies in majority of the timeframe. Although, this difference in behaviour could not be explained. In a similar study by Soro et. al. [8], the authors investigate how similar the visibility is, of different darknets varying according to the IP range, size and location. It was observed that the largest amount of scan traffic is produced by only a few source IP addresses, while most of them send only few packets. The behaviour of sources was particularly similar across darknets for UDP traffic. The study confirmed well-known facts about darknet visibility, such as the prevalence of traffic to the ports usually targeted by scans and attacks. It also provided new evidences that sources of traffic significantly varies according to the IP range, and the size of the darknet impacts its visibility.

Another prominent research domain has been based around images found from crawling the darknet, and their classification to detect and prevent nefarious activities online. Fidalgo et. al. [9] proposed a new algorithm, Semantic Attention Keypoint Filtering, to classify suspicious images in Tor darknet. The SAKF algorithm uses the Bag of Visual Words model, and image signature saliency map algorithm. The two working together, are able to extract and give importance to the foreground of the image. The SAKF algorithm achievied improvements over the baseline results, but the its effectiveness depends on the selected blurring parameter and, therefore, it does not always outperform the baseline results. Blanco-Medina et. al. [10] used a deep learning approach to improve text recognition in Tor darknet images. They tackled the problems of low resolution and skewed orientation by using Super-Resolution and Rectification respectively. The authors chose the ASTER framework as a baseline reference, and then use Residual Dense Networks (RDN), Deep CNN with Skip Connecion (DCSCN), and 4 Neural Enhance (NE) models for a total of 6 possible resolution enhancement models. All models were tried with and without rectification. Without rectification, the DCSCN scored the highest, while NE models performed best when tried with rectification.

When focus shifted to real large-scale traffic, which is diverse and complex, reduction of high false alarm rate / false positive rate (FPR) quickly became a notable issue in, with many works employing different methods to retain accuracy while constraining FPR.The CMM method, proposed by Luoshi et al. [11], based on clustering and classifier integration, was successful in reducing FPR on large-scale SSH traffic. Latah et al. [12] had explored a similar feature division and hybridisation method on unencrypted traffic, and achieved a 0.3% FPR when trained and tested on the NSL-KDD dataset, a data set suggested to solve some of the inherent problems of the KDD'99 data set which are mentioned in [13].

To the best of our knowledge, this is the first IDS model which classifies encrypted traffic using a 2-step mechanism, and utilizing the separate advantages of flow and packet based traffic features in this unique way.

2.2 Adversarial Attacks

In a darknet context, with machine learning based models now being the forefront in Intrusion Detection Systems, a huge vulnerability in the form of adversarial cyber attacks poses severe consequences; also known

as Adversarial Machine Learning (AML). This refers to a model being confused into giving an incorrect output by feeding it a benign-looking sample, which in reality contains a series of careful minute pertubations, enough to make it look unsuspecting to the human eye. AML is a harsh reality in several domains, however intrusion detection is especially important. If an IDS is exploited with this technique, it would lead to serious compromises on security and privacy.

One area of research in this domain centers around designing stronger attacks to test existing defence techniques. E. Anthi et al. [14] exhibit how samples of the Jacobian Saliency Map Attack [15] when tested on J48 and Random Forest Classifier models, lowered detection accuracy by upto 11%. Post Adversarial Training, accuracies of both models shot up, demonstrating the effectiveness of Adversarial Training as a defence mechanism, as after training the model with adversarial samples generated from 10% of the training data, results improved significantly. It also was concluded that the J48 model was not as robust against the attacks as RF. In addition to the classic more well-known attacks, studies like this one by Chen et. al. [16] formulate the generation of adversarial samples to attack Deep Neural Networks (DNNs) as an elastic-net regularized optimization problem. The same loss function is used, as in the Carlini Wagner attack [17], and the authors further propose to use the iterative shrinkage-thresholding algorithm (ISTA) [18]. Experimental results on MNIST, CIFAR10 and ImageNet showed that the L_1-based adversarial examples crafted by Elastic-Net Attack on DNN's (EAD) algorithm, could be as successful as the state-of-the-art L_2 and L_∞ attacks in breaking undefended and defensively distilled networks. Furthermore, it was concluded that EAD could improve attack transferability and help augment adversarial training.

Sadeghzadeh et. al. [19] proposed a method to attack ML/DL models using Adversarial Network Traffic (ANT), which uses uses Universal Adversarial Perturbation (UAP) to inject small pertubrations into traffic. Analysis was done on three levels - packet classification, flow content classification, and flow time series classification, with 3 different types of attacks - *AdvPad, AdvPay, and AdvBurst*. Results showed that all 3 attacks caused the classifiers to have a significant reduction in their prediction accuracy.

Usama et. al. [20] proposed a black-box adversarial attack scheme, using the Mutual Information (MI), to detect the most discriminative features. These would imply the top n features for some n, with the highest values of MI. The selected features were perturbed, and the results showed that deep learning based frameworks are gravely vulnerable to such adversarial attacks, and the authors recommend against using them in networking domains requiring a high security level.

Another prominent area of interest is Generative Adversarial Networks (GANs) [21]. GANs are used in tasks such as image-to-image translation, and in generating realistic photos of objects, scenes, and people and even deepfakes that are not discernible as fake to the human eye. Xiao et. al. [22] were the first ones to use GANs to generate adversarial samples, with their framework AdvGAN [23]. The AdvGAN framework consists of three parts: a generator, a discriminator, and the target neural network. Once the generator is trained, it can generate perturbations efficiently for any instance, so as to potentially accelerate adversarial training as defenses. The framework performed semi-white box and black box attacks with great success in both.

Since then, this methodology has only gained popularity. Li et. al. [24] proposed their GAN-based framework FlowGAN, based on the idea of dynamically altering incoming censored traffic, using features automatically learned from uncensored traffic flow features. A new metric, namely ϵ-indistinguishability was also proposed, and the attacking scheme stood comparable to other state-of-the-art attacks. Chauhan et. al. [25] propose their model Wasserstein-GAN (WGAN), which is easier to train than regular GANs, and utilizes schemes such as changing the number of features in a malicious flow, and swapping features with unused ones, to confuse IDS models which are not trained for the same. The same adversarial data was then utilised in Adversarial Training, to make the IDS robust. Usama et. al. [26] proposed a framework using GANs, which successfully evades black-box IDS's. This novel technique is able to confuse the IDS, while simultaenously ensuring the preservation of functional behavior of the network traffic features. The authors also propose a GAN-based Adversarial Training method, to overcome the disadvantage of AT only being effective against the attack samples it was trained against. The results of the study highlighted that the framework was robust against various types of adversarial attacks.

Several studies have focused on innovating different defense mechanisms, and tried to make improvements upon existing ones. For example, in an IDS context, Pawlicki et. al. [27] were the first to use neural network layer activations, and feed it into an adversarial detector, to be used in tandem with an existing ANN based IDS. Four different attacks, namely Fast Gradient Sign, Basic Iterative Method, Carlini and Wagner attack, and Projected Gradient Descent were tested. The test time neural activations of the trained ANN, and the neural activations of the adversarial examples crafted for this ANN were collected. Using these activations, the authors trained and tested five different ML classifiers to detect adversarial examples. Although the results were very

promising, the authors take note that high FPR is a sizeable hindrance, and needs to be addressed before this methodology could be further developed.

Hashemi et. al. [28] suggest a method somewhat similar to Adversarial Training, named Reconstruction from Partial Observation (RePO), wherein they use random masking in the input, and train the model with both masked and non-masked samples. They also make use of denoising autoencoders. Both of these make the NIDS non-deterministic, making it harder to craft adversarial samples against, and also avoids over-generalization. When used with the NIDS, the RePO framework, average accuracy was able to improve in both adversarial and non-adversarial contexts.

Studying the effectiveness of adversarial attacks in an Network IDS (NIDS) within a Software Defined Network (SDN), Aiken et. al. [29] implement an anomaly-based IDS *Neptune*, as the target ML-based classifier. In addition, the authors developed *Hydra*, an adversarial testing tool, to test the aforementioned effectiveness on *Neptune*. In the experiment, selected features of samples of SYN Flood DDoS traffic were perturbed by *Hydra*. K-Nearest Neighbours proved to be the most robust classifier against the adversarial attacks performed. In contrast, Random Forest, Logistic Regression, and Support Vector Machine were generally vulnerable to the same perturbations.

In encrypted traffic specifically, an interesting study by Addesso et. al. [30] tried emulating an adversary's perspective, and investigated adversarial attacks in transmitting concealed VoIP traffic data. The VoIP traffic is manipulated and concealed by using padding and shifting operations, so as to make them undetectable, resembling normal internet traffic. The authors obtain a representation of of VoIP streams in terms of transmission frequency and payload, and further characterize the range of these two properties achievable by an adversary by transforming the traffic. Lastly, the authors utilise game theory to find the Nash equilibrium of the detector and attacker schemes.

3 Motivation

In the 21st century, it is hard to find a device that is not connected to the internet. These devices could be sensors, tablets, smartphones, IoT devices, et cetera, and they all contribute to the already huge, and still growing, amount of internet traffic. Being connected to the internet, also means that by default, these devices are vulnerable to several different kinds of cyber attacks, which could have varying effects, but all stem from the compromised privacy of the user. This has led a high number of users to encrypt their traffic using tools like Tor or VPN.

Traffic that has been re-routed using these techniques, also referred to as encrypted darknet traffic, comes with its own set of challenges. The increased security at both ends also means that it is difficult for detectors to identify the origin source of encrypted traffic flows. Enemies are easily able to take advantage of this level of anonymity and execute high-level cyber attacks and ransomware attacks. This is why it is important to be able to accurately separate malicious encrypted traffic flows, from those originating from benign sources. In a traffic classification context, high FPR has also been a pressing issue that several studies have tried to address, such as T.Pietraszek' s study in 2004, which use Adaptive Alert Classification [31], and more recent studies like Grill et. al. who attempt to reduce FPR using local adaptive multivariate smoothing [32].

Due to problems like port hopping [33] and port multiplexing, which is the process of combining two or more data streams into a single physical connection, traditional traffic classification studies, which were based only on traffic flow and data packet details, have been rendered ineffective. The focus now shifts to Machine Learning/ Deep Learning based Intrusion Detection (ID) systems. These models were efficient, and accurate, but as any other ML/DL model does, they too face a vulnerability in the form of Adversarial Machine Learning (AML) [34] in which an enemy strategically crafts samples which look unsuspecting to the human eye, but are successful in confusing the models, and have them output incorrect classification results. Malfunctioning IDS models could have grave effects if they fail to detect cyber attacks in time. It could lead to huge amount of data breaches, leaks, hacks, and other compromises. This is why studying the robustness of ML/DL models in general, and IDS models in particular, is a need of the times. It is also just as crucial to test out different defense mechanisms against the aforementioned attacks.

As mentioned in the above section, we extend the existing hybrid model frameworks to an encrypted traffic IDS context in this study. We use a strategic feature division which segregates input data into two categories - flow-based data and packet-based data, to be fed into two separate ML models, and their results combined for

the final output. Latah et. al [12], showed that this approach can reduce FPR while keeping accuracy intact, for the NSL-KDD dataset, which contains a total of 39 different attacks. We take this one step further, using the CIC-Darknet 2020 dataset, which contains traffic tunnelled through both Tor and VPN, for a much more relevant, modern context.

Some of the major defense mechanisms against AML include Adversarial Training (AT) [35] [36], Neural Network Ensembles [37], and Generative Adversarial Networks (GAN's) [21]. These three defenses are summarised concisely by Short, Pay and Gandhi in [38], as : *While adversarial training has a low barrier of entry, it can be defeated by robustness of an attack. Ensembles seem to work very well (Microsoft Defender), but also have drawbacks in amount of time to develop, real time forecasting delays, and applicability of model variety (not all models will work for all types of problems and input data). Ensembles have not been tested thoroughly against the most sophisticated attacks. The GAN defense seems new and promising, as it has not been yet defeated on an academic level, but is tricky to develop and implement properly.*

We limit our focus, in this study, to Adversarial Training, and aim to propose a model that is sufficiently robust to different kinds of attacks, aided by its hybrid 2-step structure.

4 Design Component

4.1 Dataset Overview

We use the CIC-Darknet 2020 dataset for our study. This dataset is a combination of the ISCXTor2016 and ISCXVPN2016 datasets [39], and contains encrypted traffic from Tor and VPN both. We use the first layer, which classifies samples into benign and malicious traffic. The dataset contains 24,311 malicious encrypted traffic samples and 134,348 benign ones.

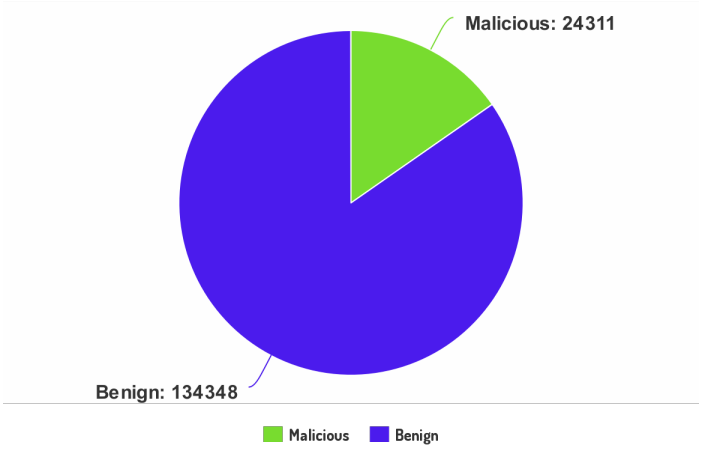

Figure 1: Distribution of samples in the CIC-Darknet 2020 dataset

The dataset contains 83 features and 2 target variable columns. We will be using the 'Label' target variable, which initially has 4 classes but is remapped as shown below :

Original	**Remapped**
Non-Tor NonVPN	Benign
Tor VPN	Malicious

Table 1: Re-labelling of target column

We drop the 'Label.1' target column which corresponds to multi-class classification of the darknet traffic into different categories based on applications used to generate the traffic. Of the 83 features, we drop 4 - 'Flow ID', 'Src IP', 'Dst IP', and 'Timestamp'. Based on manual inspection, we then separate the remaining features into 2 categories - flow-based features and data-packet based features. Table 2 shows all features contained in each category.

Flow	Src Port, Dst Port, Protocol, Flow Duration, Flow Bytes/s, Flow IAT Mean, Flow IAT Std, Flow IAT Max, Flow IAT Min, Fwd IAT Total, Fwd IAT Mean, Fwd IAT Std, Fwd IAT Max, Fwd IAT Min, Bwd IAT Total, Bwd IAT Mean, Bwd IAT Std, Bwd IAT Max, Bwd IAT Min, Fwd Header Length, Bwd Header Length, Down/Up Ratio, Subflow Fwd Bytes, Subflow Bwd Bytes, Fwd Seg Size Min, Active Mean, Active Std, Active Max, Active Min, Idle Mean, Idle Std, Idle Max, Idle Min
Packet	Total Fwd Packet, Total Bwd packets, Total Length of Fwd Packet, Total Length of Bwd Packet, Fwd Packet Length Max, Fwd Packet Length Min, Fwd Packet Length Mean, Fwd Packet Length Std, Bwd Packet Length Max, Bwd Packet Length Min, Bwd Packet Length Mean, Bwd Packet Length Std, Flow Packets/s, Fwd PSH Flags, Bwd PSH Flags, Fwd URG Flags, Bwd URG Flags, Fwd Packets/s, Bwd Packets/s, Packet Length Min, Packet Length Max, Packet Length Mean, Packet Length Std, Packet Length Variance, FIN Flag Count, SYN Flag Count, RST Flag Count, PSH Flag Count, ACK Flag Count, URG Flag Count, CWE Flag Count, ECE Flag Count, Average Packet Size, Fwd Segment Size Avg, Bwd Segment Size Avg, Fwd Bytes/Bulk Avg, Fwd Packet/Bulk Avg, Fwd Bulk Rate Avg, Bwd Bytes/Bulk Avg, Bwd Packet/Bulk Avg, Bwd Bulk Rate Avg, Subflow Fwd Packets, Subflow Bwd Packets, FWD Init Win Bytes, Bwd Init Win Bytes, Fwd Act Data Pkts

Table 2: Categorisation of features into flow-based and packet-based

In this method we end up with 2 subdivisions of the dataset, the flow-based data (hereafter referred to as FBD), and packet-based data (hereafter referred to as PBD). The FBD and PBD will be treated separately from this point on.

4.2 Model Architecture

The model, as detailed in the flowchart (Fig. 2) follows a 2-step process for traffic classification. The FBD is first fed into the flow-based classifier (CF). If this model decides that the given traffic sample is benign, it is assigned as benign and the classification process ends. If this model decides that the sample is malicious, then the PBD of that sample is further passed on to the packet-based classifier (CP), which then makes the final decision on whether the sample is benign or malicious. In this way, a traffic sample must be confirmed malicious by both CF and CP to be considered a malicious sample overall. It is this double-checking that is the key to keeping the FPR to a minimum.

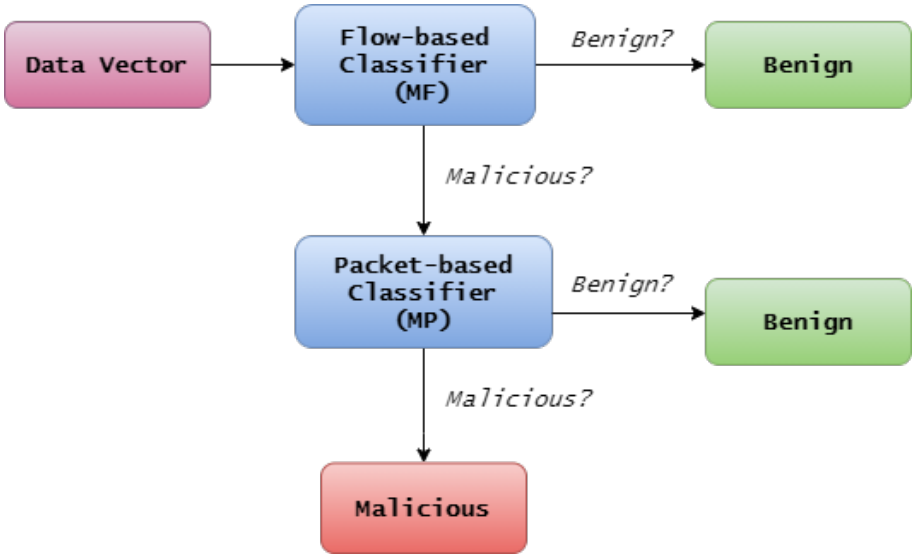

Figure 2: Architecture of Proposed Hybrid Model

4.3 Adversarial Attacks

The model is tested on adversarial samples generated by various types of attacks. These are samples generated by providing small perturbations that are imperceptible to humans, but are sufficient to confuse the model into make a wrong prediction with high confidence. [23] These adversarial samples are considered a serious issue in the deployment of ID Systems.

Adversarial attack frameworks are of three kinds - white box, black box, and grey box. The distinguishing factor is the amount of knowledge the adversary has. In white box attacks, we assume the adversary has full knowledge about the model structure and parameters. This enables the adversary to generate attacks using any means they find suitable. In grey box attacks, the only knowledge available is the structure of the model. In black-box attacks, the adversaries can only resort to the query access to generate adversarial samples. In this study, we study the robustness of our model against four such proposed black-box attacks - Fast Gradient Sign Method [40], Basic Iterative Method [41], Decision Tree Attack[42], and Jacobian Saliency Map Attack [43]. We take advantage of the transferability of these attacks [42] [44].

Fast Gradient Sign Method

This attack scheme, proposed by Goodfellow, et al., is an untargeted one-step attack scheme, which works by perturbing the sample with a vector whose elements are equal to the sign of the elements of the gradient of the cost function with respect to the input. In this way, the perturbation is obtained by linearizing the cost function of the network around the current value of the parameters of the model. The perturbation η can be expressed as:

$$\eta = \epsilon \times sign(\nabla_X J(\theta, \boldsymbol{X}, y))$$

where ϵ is a hyper-parameter which represents the size of the attack step, i.e. magnitude of perturbation, J is the cross-entropy cost function of the model, θ is the parameters of the model, $\boldsymbol{X}$ is our input, and y is the targets associated with $\boldsymbol{X}$.

Our adversarial sample $\boldsymbol{X}_{adv}$ is equal to $\boldsymbol{X} + \eta$.

Basic Iterative Method

Kurakin et. al. introduce a way to extend the FGSM, by applying it iteratively with a small step size, and clipping off values at a certain after each step to ensure that the adversarial samples lie in the ϵ neighbourhood of the original point. The number of iterations N is a user-defined value.

Mathematically, the generation of adversarial samples can be written as:

$$\boldsymbol{X}_0^{adv} = \boldsymbol{X}, \quad \boldsymbol{X}_{N+1}^{adv} = \mathrm{Clip}_{X,\epsilon}\left\{\boldsymbol{X}_N^{adv} + \alpha\,\mathrm{sign}\left(\nabla_X J\left(\boldsymbol{X}_N^{adv}, y_{\mathrm{true}}\right)\right)\right\} \tag{1}$$

Here, $\alpha T = \epsilon$, and α is the perturbation introduced in each iteration. $\mathrm{Clip}_{X,\epsilon}\left\{\boldsymbol{X}'\right\} -$ is a function which performs per-feature clipping of the adversarial sample $\boldsymbol{X}'$ after each iteration, so that it is ensured to remain in the L_∞ ϵ -neighbourhood of the original input $\boldsymbol{X}$.

Decision Tree Attack

Papernot et. al. propose a crafting algorithm targeted at decision trees, but whose adversarial samples are transferable to other models. Given an input sample and a tree, the proposed algorithm searches for leaves with incorrect classes, i.e., classes which are not the decision tree's prediction for the input. Using the path from the original leaf to the adversarial leaf, we modify features and introduce perturbations so as to force the model to mis-classify the sample in the incorrect class the newly identified leaf belongs to. The authors algorithmically define the generation as follows:

Algorithm 1: Generating Adversarial Samples in Decision Tree Attack

Input: $T, \vec{x}$, legitimate_class
$\overrightarrow{x^*} \leftarrow \vec{x}$
legit_leaf $\leftarrow$ find leaf in T corresponding to $\vec{x}$
ancestor $\leftarrow$ legitimate_leaf
component $s \leftarrow []$
while predict $\left(T, \overrightarrow{x^*}\right)$ == *legitimate_class* **do**
  if *ancestor == ancestor.parent.left* **then**
    advers_leaf $\leftarrow$ find leaf under ancestor.right
  else if *ancestor == ancestor.parent.right* **then**
    advers_leaf $\leftarrow$ find leaf under ancestor.left
  components $\leftarrow$ nodes from legit_leaf to advers_leaf
  ancestor $\leftarrow$ ancestor.parent
end
for $i \in$ *components* **do**
  perturb $\overrightarrow{x^*}[i]$ to change node's condition output
end
return $\overrightarrow{x^*}$

Jacobian Saliency Map Attack

Another scheme proposed by Papernot et. al. is the Jacobian-based Saliency Map Attack. It uses the concept of a minimum distance to travel from $\boldsymbol{X}$ such that the model mis-classifies the output. A saliency map, initially introduced as a visualization tool [45], is a map indexing all the features, whose individual values depend on how salient that feature is, which means how likely a sample with that feature perturbed is to confuse the model.

The attack proceeds in 3 steps :

1. Compute the forward derivative of $\boldsymbol{X}$. It is given by

$$\nabla \mathbf{F}(\mathbf{X}) = \frac{\partial \mathbf{F}(\mathbf{X})}{\partial \mathbf{X}} = \left[\frac{\partial \mathbf{F}_j(\mathbf{X})}{\partial x_i} \right]_{i \in 1..M, j \in 1...N}$$

where $\boldsymbol{X}$ is an M-dimensional input, and $\boldsymbol{F(X)}$ is an N-dimensional function learned by the DNN during training.

2. Construct the saliency map, which indicate to the adversary which key features to perturb. For an incorrect prediction class $\boldsymbol{t}$, input $\boldsymbol{X}$, and input feature $\boldsymbol{i}$, we aim to increase the probability of the model predicting class $\boldsymbol{t}$ while simultaneously decreasing the probability of the model predicting any other class $\boldsymbol{j \neq t}$.

The first condition rejects input components with a negative target derivative or an overall positive derivative on other classes. The product on the second line allows us to consider all other forward derivative components together in such a way that we can easily compare $S(\mathbf{X}, t)[i]$ for all input features.

$$S(\mathbf{X}, t)[i] = \begin{cases} 0 \text{ if } \frac{\partial \mathbf{F}_t(\mathbf{X})}{\partial \mathbf{X}_i} < 0 \text{ or } \sum_{j \neq t} \frac{\partial \mathbf{F}_j(\mathbf{X})}{\partial \mathbf{X}_i} > 0 \\ \left(\frac{\partial \mathbf{F}_t(\mathbf{X})}{\partial \mathbf{X}_i} \right) \left| \sum_{j \neq t} \frac{\partial \mathbf{F}_j(\mathbf{X})}{\partial \mathbf{X}_i} \right| \text{ otherwise} \end{cases}$$

High values of $S(\mathbf{X}, t)[i]$ correspond to input features that will be beneficial to the adversary by decreasing the prediction probability of the correct class, increasing the same for the other incorrect classes, or both.

3. The algorithm takes two hyperparameters, θ, and γ, which correspond to the amount a selected feature is perturbed by, and the maximum distortion permitted from the input (which is equivalent to the maximum number of distortions), respectively. If the perturbation on the selected set of features has not affected the model's performance, another set of features is selected and a new iteration occurs until a saliency map appears which can be used to generate an adversarial sample.

4.4 Metrics

To evaluate the performance of our model, we will use 4 metrics- namely Accuracy, False Positive Rate (FPR), Precision, and Recall.

Let True Positives (TP) be the number of malicious samples correctly classified; True Negatives (TN) be the number of benign samples correctly classified; False Positives (FP) be the number of benign samples mis-classified as malicious; and False Negatives (FN) be the number of malicious samples mis-classified as benign. Then, we can define our metrics as :

Accuracy $= \frac{TP+TN}{TP+TN+FP+FN}$

FPR $= \frac{FP}{FP+TN}$

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

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

4.5 Empirical Robustness

Empirical Robustness (ER)[46] is defined as the shortest distance between $\boldsymbol{x}$ and an input $\boldsymbol{x_0}$ to which classifier $\boldsymbol{f}$ assigns a label different from $\boldsymbol{f(x)}$. For a deep classifier, in binary cases, at each iteration, $\boldsymbol{f}$ is linearized around the current point $\boldsymbol{x_i}$ and the minimal perturbation of the linearized classifier is computed as

$$\arg\min_{\boldsymbol{r}_i} \|\boldsymbol{r}_i\|_2 \text{ subject to } f(\boldsymbol{x}_i) + \nabla f(\boldsymbol{x}_i)^T \boldsymbol{r}_i = 0 \tag{2}$$

The algorithm stops when we reach an $\boldsymbol{x_i}$ that is mis-classified. If the classifier function is linear, then $\boldsymbol{r_i}$ can be simply written as the orthogonal distance separating $\boldsymbol{xo}$ from the plane with the equation $F = \left\{\boldsymbol{x} : \boldsymbol{w}^T\boldsymbol{x} + b = 0\right\}$, which can be simplified to $-\frac{f(\boldsymbol{x}_0)}{\|\boldsymbol{w}\|_2^2}\boldsymbol{w}$, where $\boldsymbol{\omega}$ represents the vector of weights.

For a general classifier, its ER $\hat{\rho}_{\text{adv}}(f)$ can be represented mathematically, as:

$$\hat{\rho}_{\text{adv}}(f) = \frac{1}{|\mathcal{D}|} \sum_{\boldsymbol{x} \in \mathcal{D}} \frac{\|\hat{r}(x)\|_2}{\|x\|_2} \tag{3}$$

Where D is the set of points which have been mis-classified, and $\hat{\boldsymbol{r}}(\boldsymbol{x})$ is the l_2 norm distance of the difference between the data point and its corresponding adversarial sample, for each $\boldsymbol{x}$ in D

5 Implementation

5.1 Platform and Libraries

All implementation was done on Google Colab Pro, with our virtual machine using NVIDIA P100 GPUs and 25GB RAM. Pre-processing of data was done using the Pandas and Numpy libraries, and models were implemented using the Scikit Learn module [47] . Adversarial samples were generated using IBM's Adversarial Robustness Toolbox [48].

5.2 Data Handling

After the division of features to produce the FBD and PBD, both were scaled using the MinMaxScaler from Scikit Learn's Preprocessing module. As shown in Fig. 3, the data sets are divided into 80% training and 20% testing. For adversarial, 20% of the training data is used to generate adversarial samples, and is recombined with the untouched 80% of train data, and fed to the model for training. We use the entire test set to generate adversarial test samples. As will be discussed further on in detail, we will study the results before and after adversarial training, on both benign and adversarial test samples, to see its effect.

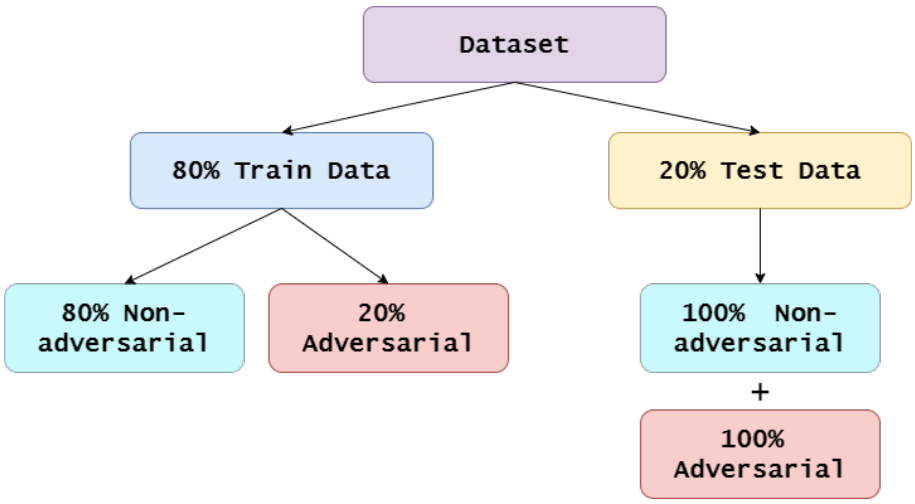

Figure 3: Division of data for adversarial training

6 Results

6.1 Non-adversarial Traffic Classification

We will use the DIDarknet framework [49] as our baseline and compare the results for CF and CP respectively. 3 different ML models, namely Random Forest [50], K-Nearest Neighbors [51], and Decision Tree [52] were tried against the baseline for both CF and CP. The results are as shown in Tables 3 & 4.

Model	Accuracy	FPR (%)	Precision	Recall
RF	0.9648	4.58%	0.9566	**0.9752**
KNN	0.9121	2.46%	0.9802	0.9493
DT	**0.9667**	**2.09%**	**0.9811**	0.9559

Table 3: Results on Non-adversarial Traffic Classification for CF

Model	Accuracy	FPR (%)	Precision	Recall
RF	**0.9710**	**1.87%**	0.9831	**0.9632**
KNN	0.9622	3.24%	0.9704	0.9572
DT	0.9701	2.67%	**0.9843**	0.9621

Table 4: Results on Non-adversarial Traffic Classification for CP

As we can see, the Decision Tree model dominates 3 out of 4 metrics for CF, and the Random Forest model does the same for CP. We also observe that on average, the models tried out for CP give better results than those for CF, but the recalls and FPR are comparable, which means CF is an appropriate first-level filter to pass on suspicious traffic samples to the more accurate CP.

We try out the hybrid model as discussed above, using a DT and RF model for CF and CP respectively. The results are as follows:

Model	Accuracy	FPR (%)	Precision	Recall
DIDarknet	0.86	-	0.86	0.86
Hybrid	0.9797	0.162%	0.9913	0.9307

Table 5: Results on Non-adversarial Traffic Classification for CP

We can see that our hypothesis was proved correct during testing, as we are able to lower FPR right down to $< 0.2\%$, while maintaining accuracy $> 95\%$.

6.2 Effect of Adversarial Training

Adversarial Training is a well known technique to improve robustness of models against cyber attacks. It solves a min-max optimization problem, with the inner maximization generating adversarial examples by maximizing the classification loss, and the outer minimization finding model parameters by minimizing the loss on adversarial examples generated from the inner maximization [On the Convergence and Robustness of Adversarial Training]. Training the model with a small set of adversarial data generated from the training set greatly improves accuracy against adversarial samples. The effect of adversarial training in an image classification context is discussed in detail in [53]. We aim to demonstrate the same on these encrypted traffic samples with our hybrid model, against our baseline DeepImage [49].

	Accuracy		FPR (%)	
	Before AT	**After AT**	**Before AT**	**After AT**
Decision Tree Attack	0.855	0.962	0.643	0.334
Fast Gradient Sign Method	0.875	0.959	0.588	0.275
Basic Iterative Method ($\epsilon = 0.1$)	0.882	0.964	0.427	0.244
Basic Iterative Method ($\epsilon = 0.01$)	0.902	0.974	0.311	0.209

Table 6: Effect of Adversarial Training in Action

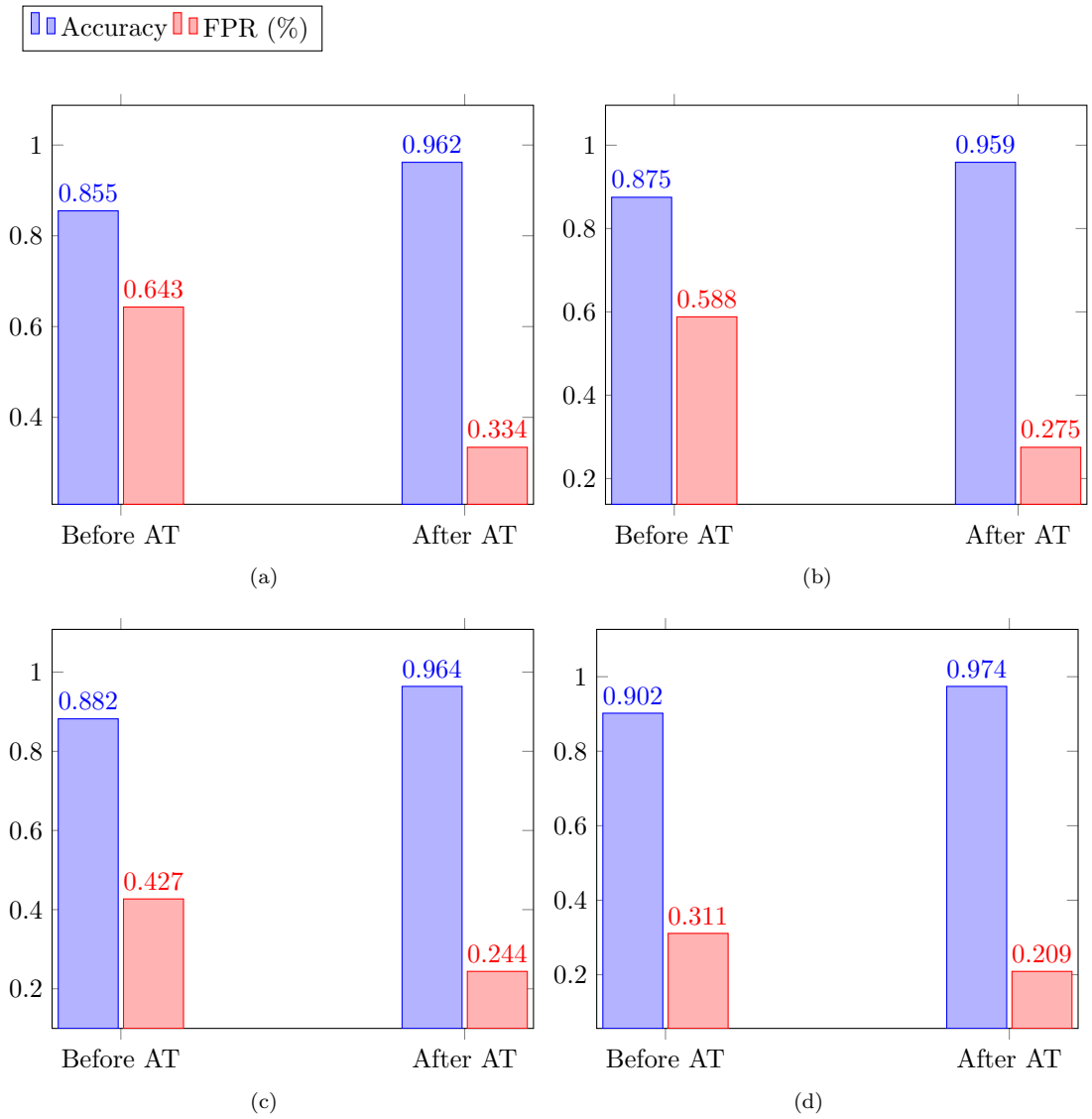

Figure 4: Tabular data visualized (a) Decision Tree Attack (b) Fast Gradient Sign Method (c) Basic Iterative Method ($\epsilon = 0.1$) (d) Basic Iterative Method ($\epsilon = 0.01$)

6.3 JSM Attack

Since the JSM Attack is a 3-step process, it is markedly slower and computationally expensive. However, compared to other attacks which alter each feature, JSMA is a more complex and elaborate approach which represents more realistic attacks as it progressively alters a small percentage of features at a time [e andhi et al]. Hence, this approach allows adversaries far more flexibility over their attacks, and makes for more real adversarial samples. Hence it is important to investigate it in detail. As mentioned above, there are 2

hyperparameters associated with the JSMA - θ, which is the perturbation to be produced in selected features, and γ, which is the maximum distortion permitted, equivalent to the number of iterations of the algorithm.

We vary both hyperparameters, γ going from 0.1 to 1, and θ going from 0.05 to 0.5, to ultimately generate a 10 x 10 map to see the effect of adversarial training on different metrics. The observed patterns will help us gain insight into the functioning of the attack algorithm against adversarial training.

Accuracy

Before AT, there is an average accuracy of 0.8683, a drop of 11.14% from the accuracy on detecting non-adversarial samples. $\theta = 0.25$ appears as a differentiating point in performances, as the average accuracy is 0.8796 for $\theta <= 0.25$ and 0.857 after, a 2% difference between the two sections.

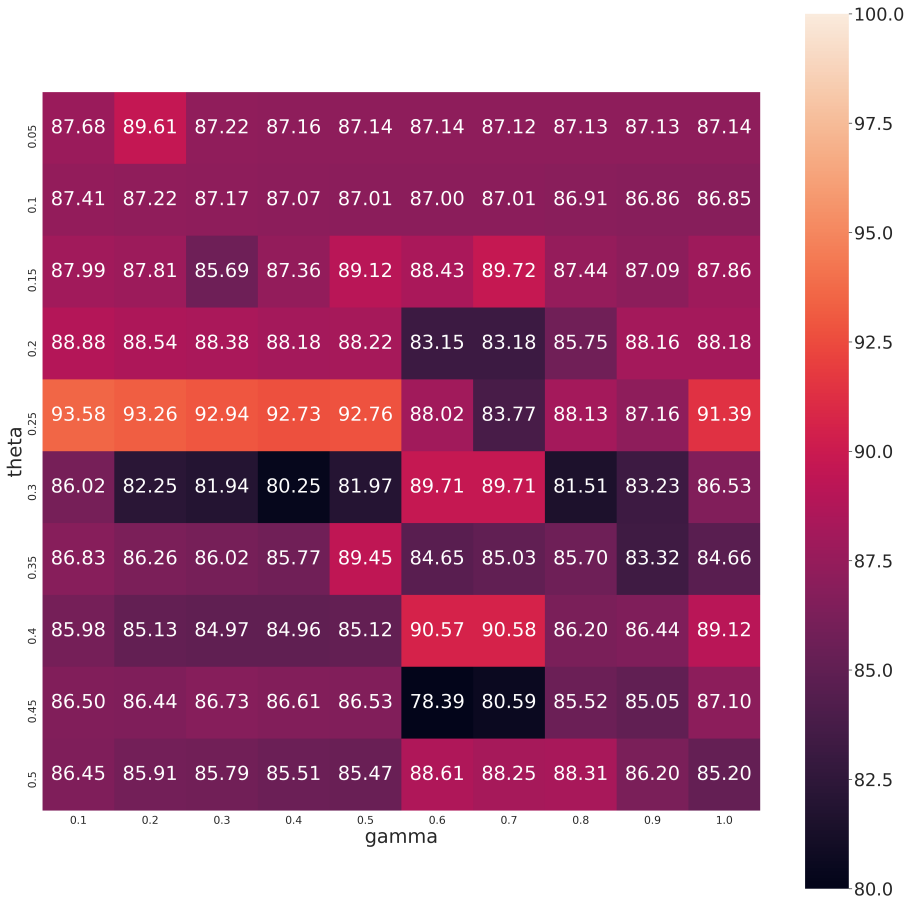

Figure 5: Testing accuracy against JSMA samples before AT

After AT, there is a stark improvement in accuracy - the average increases to 0.9701% with the drop reducing to just 0.96%. Any previously visible patterns have also been mellowed out, as the accuracy is stabilised across all 100 parameter settings. The stark 2% difference previously visible at the $\theta = 0.25$ mark, is now reduced to just 0.08%. This sort of stabilisation makes it significantly harder for an adversary to try and find a weak spot in tuning the attack hyperparameters.

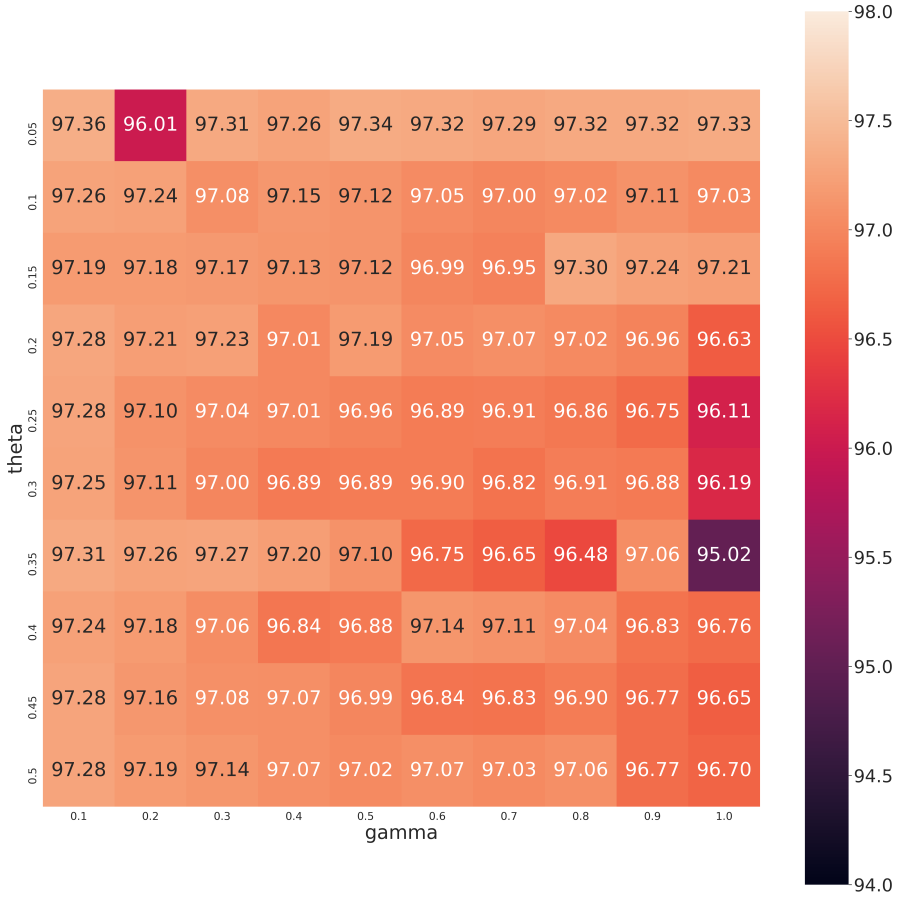

Figure 6: Testing accuracy against JSMA samples after AT

FPR

FPR appears fairly randomly distributed when the model is tested on JSMA samples, we can say it generally increases with θ, but has no visible meaningful correlation with γ.

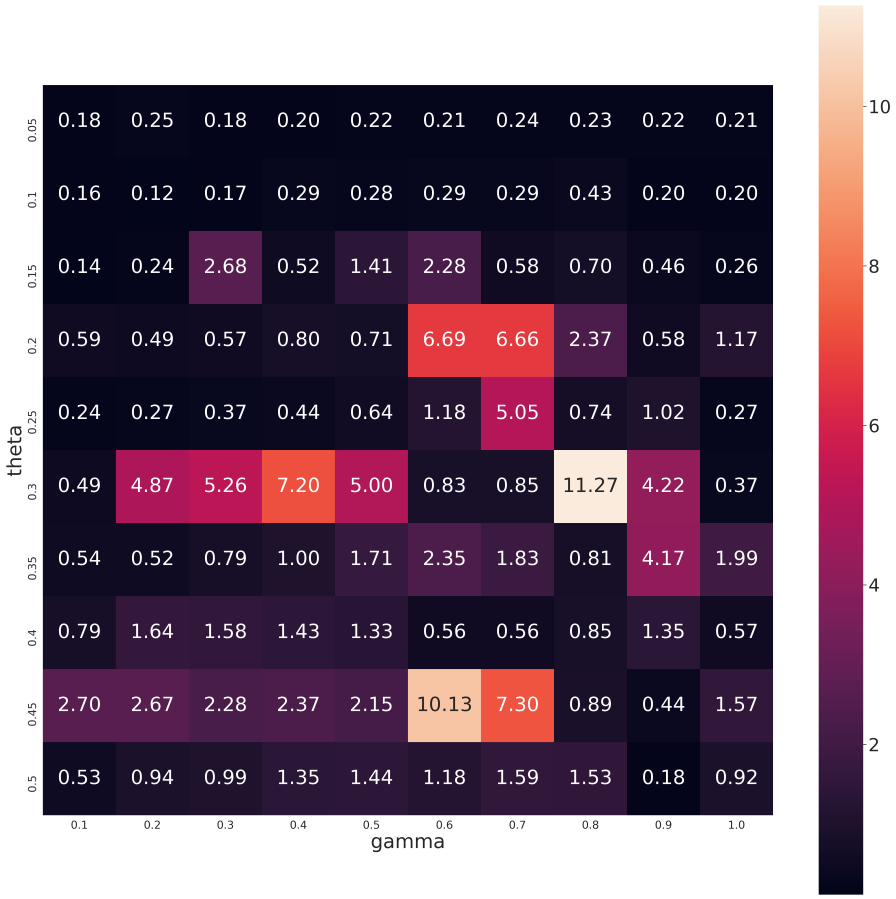

Figure 7: FPR against JSMA samples before AT

As seen in Figure 8, the earlier seen randomness no longer visible, FPR is mostly stable and never crosses 0.38%, as compared to our non-adversarial testing FPR of 0.162%. The samples of the range θ from 0.25 to 0.35 have highest average FPR by a very narrow, almost negligible margin. Correspondingly, the same rows have the 3 lowest average accuracies post-AT by a small difference.

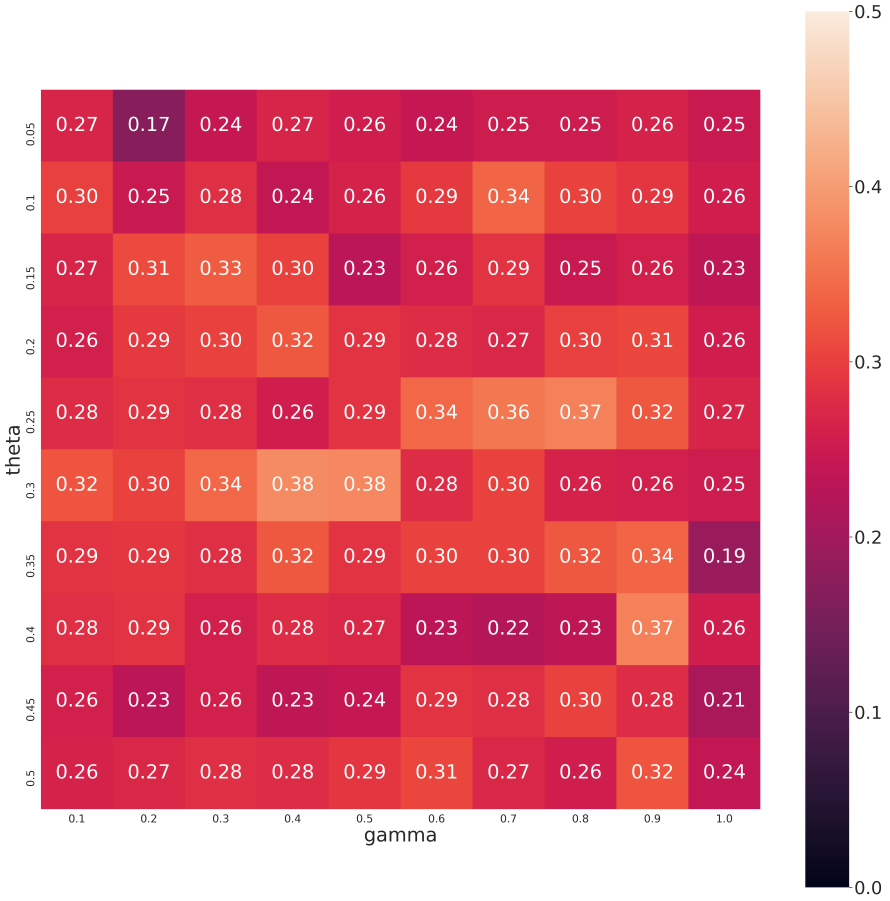

Figure 8: FPR against JSMA samples after AT

6.4 Empirical Robustness

As defined in Section 4.4, Empirical Robustness (ER) is defined as the shortest distance between $\boldsymbol{x}$ and an input $\boldsymbol{x_0}$ to which classifier $\boldsymbol{f}$ assigns a label different from $\boldsymbol{f(x)}$. This implies, naturally, that a higher value of ER is indicative of a more robust model.

As there is no pre-defined standard on how to implement this metric for multi-step models, we propose the following: In our case, as the data being fed to both models has no commonalities, and are derived from the same dataset, for the purpose of calculating ER we simply concatenate the FBD and PBD, and use it as an overall dataset, leaving any feature divisions and conditional forwarding as an implicit functioning of the model. In this way, both models will be given weightage corresponding to the vulnerability of their features, and we will be able to consider just the final output of the model, instead of using intermediary results, and producing potentially inconclusive or incorrect results.

Using this method, we can also demonstrate the advantage of our hybrid model when compared to single step models we tried in Section 6.

Model	**DT Attack**	**FGSM Attack**	**BIM ($\epsilon = 0.1$)**	**BIM ($\epsilon = 0.01$)**
Hybrid Model	**0.2653**	0.4346	**0.4341**	0.0464
KNN	0.2303	0.3909	0.3932	**0.0473**
DT	0.1445	0.3653	0.3781	0.0435
RF	0.2147	**0.4558**	0.3977	0.0470

Table 7: Comparison of Empirical Robustness (ER) against different attacking schemes

As seen in the table, our hybrid model is discernibly better for 2 of the attacks, namely Decision Tree Attack and the BIM attack with $\epsilon = 0.1$, and falls short by an acute margin for the BIM attack with $\epsilon = 0.01$. The Random Forest proves slightly more robust for the Fast Gradient Sign Method attack. But since the hybrid model outdoes the other 3 on accuracy and FPR, it is a small price to pay for a desirable result. The hybrid model is clearly advantageous when we consider the simultaneous optimization of accuracy, FPR, and ER across different adversarial attack schemes.

7 Conclusion

In this paper, we proposed a hybrid framework for an IDS, with a feature division based on flow-based and data packet-based features. Our framework deals with two pressing issues in existing ID Systems, namely keeping the False Positive Rate (FPR) low, and having it be robust against different adversarial attacks. We trained our model using a re-labelled version of the Level 1 'Label' target variable, to separate incoming traffic into benign and malicious categories. While comparing our framework with individual ML models on the same dataset, we were successful in showing its supremacy in maintaining an FPR as low as 0.162% for non-adversarial traffic classification. We then generated black-box adversarial attack samples using IBM's Adversarial Robustness Toolbox (ART). Utilisng Adversarial Training (AT) as our defense mechanism, we train the model with the generated samples with an 80-20 combination of non-adversarial and adversarial samples. Comparing the results of prediction on adversarial test samples before and after AT, we see that the drop in classification accuracy, as well as FPR, is significantly reduced. We also use Empirical Robustness (ER) as our robustness metric, and observe a stark increase in ER before vs. after AT for all four attack samples. Lastly, we take a deeper dive into the Jacobian Saliency Map Attack (JSMA), and study the effect of varying its hyperparameters γ θ, on the framework's prediction accuracy and FPR. This is in an attempt to mimic the functioning of a real adversary.

Bibliography

[1] N. Furutani, T. Ban, J. Nakazato, J. Shimamura, J. Kitazono, and S. Ozawa, "Detection of ddos backscatter based on traffic features of darknet tcp packets," in *2014 Ninth Asia Joint Conference on Information Security*, 2014, pp. 39–43.

[2] J. Liu and K. Fukuda, "Towards a taxonomy of darknet traffic," in *2014 International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2014, pp. 37–43.

[3] H. Nishikaze, S. Ozawa, J. Kitazono, T. Ban, J. Nakazato, and J. Shimamura, "Large-scale monitoring for cyber attacks by using cluster information on darknet traffic features," *Procedia Computer Science*, vol. 53, pp. 175–182, 2015, iNNS Conference on Big Data 2015 Program San Francisco, CA, USA 8-10 August 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050915017950

[4] E. Bou-Harb, "A probabilistic model to preprocess darknet data for cyber threat intelligence generation," in *2016 IEEE International Conference on Communications (ICC)*, 2016, pp. 1–6.

[5] N. Vichaidis, H. Tsunoda, and G. M. Keeni, "Analyzing darknet tcp traffic stability at different timescales," in *2018 International Conference on Information Networking (ICOIN)*, 2018, pp. 128–133.

[6] C. Han, J. Shimamura, T. Takahashi, D. Inoue, M. Kawakita, J. Takeuchi, and K. Nakao, "Real-time detection of malware activities by analyzing darknet traffic using graphical lasso," in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2019, pp. 144–151.

[7] F. Gadhia, J. Choi, B. Cho, and J. Song, "Comparative analysis of darknet traffic characteristics between darknet sensors," in *2015 17th International Conference on Advanced Communication Technology (ICACT)*, 2015, pp. 59–64.

[8] F. Soro, I. Drago, M. Trevisan, M. Mellia, J. Ceron, and J. J. Santanna, "Are darknets all the same? on darknet visibility for security monitoring," in *2019 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, 2019, pp. 1–6.

[9] E. F. Fernandez, R. A. V. Carofilis, F. J. Martino, and P. B. Medina, "Classifying suspicious content in tor darknet," 2020.

[10] P. Blanco-Medina, E. Fidalgo, E. Alegre, and F. Jáñez-Martino, "Improving text recognition in tor darknet with rectification and super-resolution techniques," in *9th International Conference on Imaging for Crime Detection and Prevention (ICDP-2019)*, 2019, pp. 32–37.

[11] Z. Luoshi, X. Yibo, and Y. Bao, "A new network traffic classification method based on classifier integration," *International Journal of Grid and Distributed Computing*, vol. 8, pp. 309–322, 06 2015.

[12] M. Latah and L. Toker, "Minimizing false positive rate for dos attack detection: A hybrid sdn-based approach," *ICT Express*, vol. 6, no. 2, pp. 125–127, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405959519303480

[13] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009, pp. 1–6.

[14] E. Anthi, L. Williams, M. Rhode, P. Burnap, and A. Wedgbury, "Adversarial attacks on machine learning cybersecurity defences in industrial control systems," *J. Inf. Secur. Appl.*, vol. 58, p. 102717, 2021.

[15] N. Papernot, P. Mcdaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 372–387, 2016.

[16] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, "Ead: Elastic-net attacks to deep neural networks via adversarial examples," 2018.

[17] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," 2017.

[18] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imaging Sci.*, vol. 2, pp. 183–202, 2009.

[19] A. M. Sadeghzadeh, S. Shiravi, and R. Jalili, "Adversarial network traffic: Towards evaluating the robustness of deep learning-based network traffic classification," 2021.

[20] M. Usama, A. Qayyum, J. Qadir, and A. Al-Fuqaha, "Black-box adversarial machine learning attack on network traffic classification," in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2019, pp. 84–89.

[21] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[22] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," 2019.

[23] K. Ren, T. Zheng, Z. Qin, and X. Liu, "Adversarial attacks and defenses in deep learning," *Engineering*, vol. 6, no. 3, pp. 346–360, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S209580991930503X

[24] J. Li, L. Zhou, H. Li, L. Yan, and H. Zhu, "Dynamic traffic feature camouflaging via generative adversarial networks," in *2019 IEEE Conference on Communications and Network Security (CNS)*, 2019, pp. 268–276.

[25] R. Chauhan and S. Shah Heydari, "Polymorphic adversarial ddos attack on ids using gan," in *2020 International Symposium on Networks, Computers and Communications (ISNCC)*, 2020, pp. 1–6.

[26] M. Usama, M. Asim, S. Latif, J. Qadir, and Ala-Al-Fuqaha, "Generative adversarial networks for launching and thwarting adversarial attacks on network intrusion detection systems," in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2019, pp. 78–83.

[27] M. Pawlicki, M. Choraś, and R. Kozik, "Defending network intrusion detection systems against adversarial evasion attacks," *Future Generation Computer Systems*, vol. 110, pp. 148–154, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X20303368

[28] M. J. Hashemi and E. Keller, "Enhancing robustness against adversarial examples in network intrusion detection systems," 2020.

[29] J. Aiken and S. Scott-Hayward, "Investigating adversarial attacks against network intrusion detection systems in sdns," in *2019 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2019, pp. 1–7.

[30] P. Addesso, M. Cirillo, M. D. Mauro, M. Longo, and V. Matta, "Adversarial detection of concealed voip traffic," in *2019 International Conference on Computing, Networking and Communications (ICNC)*, 2019, pp. 437–441.

[31] T. Pietraszek, "Using adaptive alert classification to reduce false positives in intrusion detection," in *Recent Advances in Intrusion Detection*, E. Jonsson, A. Valdes, and M. Almgren, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 102–124.

[32] M. Grill, T. Pevný, and M. Rehak, "Reducing false positives of network anomaly detection by local adaptive multivariate smoothing," *Journal of Computer and System Sciences*, vol. 83, no. 1, pp. 43–57, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022000016300022

[33] H. Lee and V. Thing, "Port hopping for resilient networks," vol. 5, 10 2004, pp. 3291 – 3295 Vol. 5.

[34] L. Huang, A. Joseph, B. Nelson, B. I. P. Rubinstein, and J. Tygar, "Adversarial machine learning," in *AISec '11*, 2011.

[35] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[36] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2019.

[37] T. Strauss, M. Hanselmann, A. Junginger, and H. Ulmer, "Ensemble methods as a defense to adversarial perturbations against deep neural networks," 2018.

[38] A. Short, T. La Pay, and A. Gandhi, "Defending against adversarial examples." [Online]. Available: https://www.osti.gov/biblio/1569514

[39] A. Habibi Lashkari, G. Draper Gil, M. Mamun, and A. Ghorbani, "Characterization of tor traffic using time based features," 01 2017, pp. 253–262.

[40] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2015.

[41] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," 2017.

[42] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," 2016.

[43] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," 2015.

[44] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," 2017.

[45] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2014.

[46] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," 2016.

[47] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[48] M.-I. Nicolae, M. Sinn, T. N. Minh, A. Rawat, M. Wistuba, V. Zantedeschi, I. M. Molloy, and B. Edwards, "Adversarial robustness toolbox v0. 2.2," 2018.

[49] A. Habibi Lashkari, G. Kaur, and A. Rahali, "Didarknet: A contemporary approach to detect and characterize the darknet traffic using deep image learning," in *2020 the 10th International Conference on Communication and Network Security*, 2020, pp. 1–13.

[50] A. Tesfahun and D. L. Bhaskari, "Intrusion detection using random forests classifier with smote and feature reduction," in *2013 International Conference on Cloud & Ubiquitous Computing & Emerging Technologies*. IEEE, 2013, pp. 127–132.

[51] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[52] S. Sahu and B. M. Mehtre, "Network intrusion detection system using j48 decision tree," in *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2015, pp. 2023–2026.

[53] S. Park and J. So, "On the effectiveness of adversarial training in defending against adversarial example attacks for image classification," *Applied Sciences*, vol. 10, no. 22, 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/22/8079