Distinguishing between Benign & Malicious traffic on IoT networks

First Author
Parth J. Patel
M. Sc. Applied Data Science
School of information Studies
Syracuse University
Syracuse, NY 13210 USA
papatel@syr.edu

Second Author
Dishank K. Solanki
M. Sc. Information Management
School of information Studies
Syracuse University
Syracuse, NY 13210 USA
dsolanki@syr.edu

Pawan SP
M. Sc. Applied Data Science
School of information Studies
Syracuse University
Syracuse, NY 13210 USA
psharda@syr.edu

Abstract
This paper focuses on trying to identify the factors which help in determining the type of traffic over a IoT network. Our findings will include analysis to determine the type of traffic which would be our first objective i.e., a binary classification task. And our secondary objective is to classify various types of attack plus benign traffic as well. i.e., Multinomial classification of 9 classes. We leverage several classification techniques to first identify what factors help in distinguishing between benign and malicious traffic and then we run a black box Deep Learning Multi-Layer Perceptron to perform the Multinomial classification task. The data was leveraged from a private real simulation of IoT devices. This allowed us to focus more on the classification task rather than initial pre-processing.

1 Introduction
Samsung is one of the companies that manufactures IoT devices for everyday use. There are several other companies which do the same such as Philips, Provision etc. Their products include cameras, doorbells, heat sensors and motion sensors. All of which can be helpful in local house security or disruptive for the same. A need is present for a monitory system which can detect malicious traffic in our everyday devices such as Security cameras, Doorbells, Thermostats even baby monitors. The problem here is that many companies are rushing out new IoT devices such as Doorbells and security cams without any said security systems. This leads to many small invasions of privacy and other extreme acts such as Identity Thefts.

Our goal is to use network traffic as a base with our domain knowledge in Cyber Security and networking to combine it with data science and create a model which can accurately identify Malicious traffic over the said IoT devices.

A **malicious attack** is an attempt to forcefully abuse or take advantage of someone's computer, whether through computer viruses, social engineering, phishing, or other types of social engineering. We are focusing on 2 types of major IoT botnet attacks:

1. Mirai Attack: Mirai is a malware that turns networked devices running Linux into remotely controlled bots that can be used as part of a botnet in large-scale network attacks. It primarily targets online consumer devices such as IP cameras and home routers. It is majorly used for DDOS attack. Different types of Mirai attacks are: ACK, SYN, UDP Plain. (Jiyeon Kim n.d.)

    i. ACK Attack (ACK Flood attack): An ACK flood attack is when an attacker attempts to overload a server with TCP ACK packets. Like other DDoS attacks, the goal of an ACK flood is to deny service to other users by slowing down or crashing the target using junk data. The targeted server must process each ACK packet received, which uses so much computing power that it is unable to serve legitimate users.
    ii. SYN Attack (SYN Flood Attack): A SYN flood, sometimes known as a half-open attack, is a network-tier attack that bombards a server with connection requests without responding to the corresponding (CloudFare n.d.)acknowledgements. The large numbers of open TCP connections that result consume the server's resources to essentially crowd out legitimate traffic, making it impossible to open new legitimate connections and difficult or impossible for the server to function correctly for authorized users who are already connected.
    iii. UDP Plain attack: It is nothing but UDP attack for higher packet per second.

2. Gafgyt attack: Also known as Bashlite is a malware which infects Linux systems to launch distributed denial-of-service attacks (DDoS). The original version in 2014 exploited a flaw in the bash shell - the Shellshock software bug - to exploit devices running BusyBox. A few months later a variant was detected that could also infect other vulnerable devices in the local network. In 2015 its source code was leaked, causing a proliferation of different variants, and by 2016 it was reported that one million devices have been infected. Different types of Gafgyt attacks are Scan, Junk, UDP, TCP, combo. (Jiyeon Kim n.d.)

    i. Scan Attack: This type of attack occurs when the software are made to scan the networks in search for vulnerable IOT devices.
    ii. Junk Attack: It is nothing but spamming the IOT devices with spam data.
    iii. UDP Attack (UDP Flood Attack): A UDP flood is a type of denial-of-service attack in which many User Datagram Protocol (UDP) packets are sent to a targeted

server with the aim of overwhelming that device's ability to process and respond. The firewall protecting the targeted server can also become exhausted because of UDP flooding, resulting in a denial-of-service to legitimate traffic. (F5 n.d.)

iv. TCP: **TCP** SYN flood (a.k.a. SYN flood) is a type of Distributed Denial of Service (DDoS) **attack** that exploits part of the normal **TCP** three-way handshake to consume resources on the targeted server and render it unresponsive. (Imperva n.d.)

v. COMBO attack: Its combination of attack of sending spamming data and sending open connection of IP and port to IOT devices. (CloudFare n.d.)

2.1 Data and Features

The data was obtained from (Y. Mirsky 2018) they were the group that performed all the preprocessing i.e., conversion from *.pcap to *.csv. They had obtained the original data from the study (Y. Meidan 2018). The data contains 4 measurements with 5 timeframes and up to 5 statistics of each. This amounted to a total of 115 feature set for each IoT interaction. The data has 7,062,606 interactions or transactions between IoT devices.

Types of Measurements: (Y. Meidan 2018)

Title	Description
H and MI	Summary of recent traffic from the packet's host (IP), MI is the same thing but for Mac address
HH	Summary of recent traffic from Packet's host to packet's destination (IP)
HpHp	Summary of the recent traffic from host + port to destination's host + port. This is to measure if different devices from same IP are attacking or no.
HH_jit	Summary of jitter of the traffic from the packet's host to destination (IP)

Types of Time frames:

Title	Description
L5	Last 5 seconds
L3	Last 3 seconds
L1	Last 1 second
L01	Last 10 milli seconds
L001	Last 1 milli seconds

Types of Statistics:

Title	Description
Weight	Weight of the stream i.e., number of items observed in the recent time frame.
Mean	Average of the summary
Std	Standard deviation of the summary
Radius or Magnitude	The root squared sum of the two streams' variances
Cov or pcc	An approximated covariance between two streams.

Nomenclature rule: X_Y_Z here X represents the above-mentioned measurements, Y represents the time frames and Z represents the type of statistic. Example: HH_L5_weight represents total number of packets sent in recent traffic from host to destination (IP) in the last 5 seconds.

2.2 Pre-processing

Although the data was thoroughly feature engineered, we had to perform some encoding initially. We added 2 new features which represents our 2 target variables, one represented either benign or malicious where 1 denotes malicious, and the 2nd feature which had encoding from 0 to 8 where 0 denoted benign attacks and 1 through 8 denoted different types of attacks which are mentioned above. We also had to randomly pick some data points as the data was highly imbalanced. We chose about 900,000 data points which amounted to 100,000 for each class. A point to be noted is that our binary classification would be performed on highly biased data with 1:9 ratio. The distribution of 2nd target variable was uniform. We also converted all the features which were strings into floats.

2.3 Models

We decided to run various models with different parameters to find out best models. For binary classification, the validation parameter was AUC, whereas for Multinomial classification the validation parameter was F1-micro-measure. The models are listed below.

Model	**Parameters**	**Task**
Logistic Regression	Regularization and Elastic net combinations: (0.1, 0.2), (0.2, 0.4), (0.3,0.5) as (Rg, Elast)	Binary class
Random Forest	Max Depth 30, Number of trees 100, and defaults	Binary class
Random Forest	Default parameters	Multi class
Multi-Layer Perceptron	Step size 0.01, Max iterations 400, layer architecture: 115, 40, 40, 15, 9.	Multi Class

Note: Every Pipeline also had a standard scaler stage.

Data was split into 3 sets: Training data 648,109, Validation data 185,393 and Testing data 92,790 with ratio 7:2:1.

3 Results

The following summarize the results obtained from the analysis:

3.1 Logistic Regression

To classify a binary task Logistic regression is perhaps the most common and simple approach. It passes the linear equation through a logistic function or sigmoid function Exp(x)/(1+Exp(x)) where x represents the linear input to generate a monotonic output i.e., 0 or 1.

We introduced our data to the logistic regression model initially with default parameters of pyspark.ml package. The results on the validation data set showed an AUC of 1. Which indicates a TPR of 100%. We presumed that the model over fit and therefore we started to introduce generalization in our model, as the goal of our analysis is to determine methodology for security of all IoT devices we needed a more generalized answer rather than a over fit one.

The first tuning stage we introduced a regularization parameter of 0.1 and Elastic net parameter of 0.2 we did not notice any change in model performance. Our goal was to make 2 more models with higher regularization to see how they fare with our validation data. We introduced regularization of 0.2 and elastic of 0.4 and another model with regularization of 0.3 and elastic net of 0.5 introducing larger penalties for higher weights and L1 and L2 of the lasso and ridge methods using elastic net.

We noticed some drop in performance in the 3rd logistic model. However, the model 1 and 2 performed very well in the validation set. Since increasing the regularization parameters and the elastic net penalty did not generalize the model on the validation data better, we decided that the model 1 with 0.1 and 0.2 parameters was the best fit and well generalized. The AUC on these 3 models were 0.999, 0.998 and 0.752, respectively.

The testing performance for the Model 1 was accuracy: 0.998, f1-score: 0.999, precision: 0.999 and recall was 0.999.

The most important features with the highest weights in the logistic regression model 1 were HH_jit at L001 and L01. This corresponds to the Jitter in the recent traffic from One host to one destination in the recent 1 millisecond and 10 milliseconds. This makes sense intuitively as when a device is being attacked it will receive a lot of data from many sources and this will cause a lot of jitter over the network. This will not be true for benign traffic. This way from the logistic regression we can understand that jitter is a very important feature. This maybe something we could recommend companies to focus on when building IoT devices.

3.2 Random Forest Classification

To classify a binary task, one of the more modern techniques is Random Forest. Mostly it uses decision trees as a base model and combines outputs from many different trees to produce a well generalized and powerful model which allows us to perform complex tasks with ease.

When the data was passed through a Random Forest model, we can observe that the performance is good with the parameters we chose. The initial model we ran was with default parameters and our tuned model with 30 max depth and 100 number of trees managed to not over fit as it had better performance on the validation data and Testing data when compared to the default model. The validation AUC was about 0.9999 which is almost 1.0. The performance on the test data set was good as well, accuracy amounted to 0.99, f1-score was 0.99, precision was 1.0 and recall was 0.99.

The features which were important in our random forest model were HH_jit, HH at 001 timeframe. We also observe some other variables also reduce the loss function such as HpHp and MI. Therefore, it is important to focus on these variables while trying to make improvements in the technology with respect to security. Host to Host jitter has been of the utmost importance in classifying Benign or Malicious attacks. The timeframe which is most prevalent in Identifying the attacks is L001.

Multinomial Random Forest: The Multinomial Random Forest did not perform well at all. The Validation F1-score was about 0.13 and the testing accuracy was about 0.22 percent. This was a model run on base parameters and trying to increase the model complexity did not help in improving the scores. Therefore, we concluded that Random Forest do not possess enough complexity to make a distinction between different types of attacks. We decided to move on from this stage to a much more suitable classifier for such a complex problem.

3.3 Multi-Layer Perceptron

The most complex model in today's machine learning world is a Neural Network with deep learning architecture. A basic Neural net is considered as a DL model when there are more than 2 hidden layers.

In our model we decided to use a 3 hidden layer architecture with about 400 maximum iterations allowed to reach optima. We included a relatively neutral step size of 0.01 this is also known as the learning rate. It can easily be understood if we imagine it like the steps of an MCMC robot. The architecture was as follows:

Layer type	Nodes	Activation
Input	115	
Dense hidden	40	Sigmoid
Dense hidden	40	Sigmoid
Dense hidden	15	Sigmoid
Output	9	SoftMax

Although the multilayer perceptron has a lot of time complexity it works well for complex problems such as this. The F1-micro measure on the validation dataset was about 0.87 and the Testing accuracy was 0.89. The model worked well when compared to the Random Forest classifier.

An unusual problem with Deep learning architecture is that we lose model interpretation. It is relatively difficult to understand which variables improve the model the most. Comparatively a model such as Logistic or Random Forest may work less efficiently but allow us to interpret the model easily with respect to change in probability and reduction of the loss function, respectively.

There are techniques which can be introduced to interpret the Deep neural networks better such as Individual Conditional Expectation (ICE), Permutation Feature importance Scores, Partial Dependence Plots (PDP), Local Interpretable Model-Agnostic Explanations (LIME), Deep Learning Important Features (DeepLIFT), SHapley Additive Explanations (SHAP) and several gradient based methods.

4 Conclusion

We chose this topic because it is a relatively unpopular domain in data science but is increasingly gaining momentum in today's day and age. Modern devices such as Alexa which me personally bought today as I type this are becoming increasingly popular in houses. It makes it easy for ill-intended third parties to attack and misuse such devices and disrupt one's life in best case and steal one's identity in worst case. Our results add some notion to approaching a solution to this issue. A method would be to focus on why jitter always turns up to be most important feature when detecting malicious traffic from benign. Our work was mainly to perform binary classification however we tried to focus on different types of attacks as well just to understand if there are some attacks which are easier to detect than others. However, the balanced results which we obtained show us that there will always be some cases where attacks will not be classified correctly but we can almost always rely on a binary classification to identify that there has been an attack. The only constraint is that we need to measure the devices every L001 timeframe because that was the most prevalent in improving the models. Lastly, this would be the perfect problem to overfit as attacks have been the same since 2016 and if we can keep detecting attacks on types of devices, we can develop technology for every device which may help in preventing fraud.

Acknowledgements

We would like to thank Professor Daniel Acuna for all the help throughout the project. We would also like to thank him for insights into Data Science theory and providing relevant examples. We would like to thank all our classmates who helped us with improving and provided valuable feedback and a lot of encouragement.

References

CloudFare. n.d. "https://www.cloudflare.com/learning/ddos/."

F5. n.d. "https://www.f5.com/services/resources/glossary/syn-flood."

Imperva. n.d. "https://www.imperva.com/learn/ddos/syn-flood/."

Jiyeon Kim, Mnsun Shim, Seungah Hong, Yulim Shin, Eunjung Choi. n.d. "Intellident Detection of IoT Botnets using ML and Deep learning."

Mikhail Khodak, Nikunj Saunshi, Kiran Vodrahalli. 2017. "https://nlp.cs.princeton.edu/SARC/0.0/." *Development version of the Self-Annotated Reddit Corpus (SA.*

Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, D. Breitenbacher, A. Shabtai, and Y. Elovici. 2018. "N-BaIoT: Network-based Detection of IoT Botnet Attacks Using Deep Autoencoders." *IEEE Pervasive Computing, Special Issue - Securing the IoT.*

Y. Mirsky, T. Doitshman, Y. Elovici & A. Shabtai. 2018. "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection." *Network and Distributed System Security (NDSS) Symposium, San Diego, CA, USA.*