Machine Learning

Project Report

Title:

DDos Botnet Attack Detection

Submitted to:

Sir Babar Hameed

Submitted By:

[2020-CE-47]

Department of Computer Engineering

University of Engineering and Technology, Lahore

DDos Botnet Attack Detection

Hira Firdous
2020-CE-47
Computer Engineering Department
University of Engineering and Technology, Lahore

Abstract

The study explores DNS-Based Botnet Detection in cybersecurity, focusing on identifying threats like DNS-based botnets and addressing challenges in analyzing records for DDoS attacks. It emphasizes the need for advanced machine learning techniques, aims to investigate models for detecting malicious traffic in IoT networks, and evaluates algorithms such as Decision Tree Classifier and Bagging Classifier. Using the IoT datasets, the research employs data exploration, preprocessing, and machine learning models, highlighting their strengths and weaknesses. Evaluation metrics include accuracy, precision, recall, F1 score, ROC, and a confusion matrix. Findings showcase the Bagging Classifier's enhanced predictive performance with 99.9% precision and recall. Limitations include training on specific files, suggesting future work on comprehensive dataset training and real-world applications, contributing to advancing machine learning models in IoT device security.

1. Introduction

1.1 Background

Within the dynamic realm of cybersecurity, the utilization of DNS-Based Botnet Detection encapsulates the intricate interplay between advancement and risk, echoing the wisdom that " the line between progress and peril is often thin, reminding us that the impact of any tool is shaped by the hands that employ it." Consequently, it becomes imperative to formulate a method for identifying diverse threats on the internet. The specific focus of this study is on DNS-based botnets. A botnet constitutes a collection of compromised internet-connected devices, each infiltrated with malware designed to manipulate it remotely, unbeknownst to the legitimate owner of the device.

1.2 Problem Statement

Analyzing the records to determine whether a computer has experienced a DDos attack necessitates extensive effort and statistical analysis. Improving and expediting this process involves refining the methodology and adding advance machine learning techniques.

1.3 Objectives

The following are this study's main goals:

- To investigate how machine learning models may be used to identify malicious traffic in Internet of Things networks.
- To get the better understanding of Machine learning algorithms like Decision Tree Classifier, Bagging Classifier and their implementation.
- Get the idea of each models Pros and cons by analyzing their pros and cons.
- Evaluating their performance on certain metrics like F1 score, recall, accuracy, and precision.
- Must carry out cross-validation in order to guarantee the models' generalization and resilience.

Literature Review

Strong security measures are required because of the Internet of Things' (IoT) explosive growth, which has greatly increased the attack surface for bad actors. The dynamic threats seen in Internet of the literature survey reveals various techniques for detecting Botnet attacks, each presenting limitations in scalability, data set size, accuracy, and processing speed. To address these challenges, the study advocates for the adoption of Machine Learning (ML) as a more viable approach. The research conducts a performance analysis of key ML methods such as Support Vector Machine (SVM), Naïve Bayes (NB), Artificial Neural Network (ANN), Decision Tree (DT), and Unsupervised Learning (USML), including K-means and X-means. The choice of these ML methods is based on their scalability advantages and appropriateness for the data under consideration. A framework is outlined for performance evaluation, encompassing components like traffic packet collection using tools like 'tcpdump,' feature generation with tools such as Bro and Argus, and the selection of genuine features through filtering methods. The ML procedures are detailed, emphasizing SVM's ability to detect both non-spoofed and spoofed IP, ANN's capacity for linear and non-linear data classification, NB's efficiency with minimal training data, DT's feature selection via a tree-like structure, and USML's application in discovering patterns within complex data related to Botnet activities [1]

Prior to model development, it is crucial to undergo data exploration and preprocessing to ensure that machine learning models can identify patterns and produce accurate predictions. The literature review underscores the significance of metrics in evaluating the efficacy of intrusion detection models, encompassing key measures such as accuracy, precision, recall, and F1 score.

3. Dataset Description

3.1 IoT-23 Dataset

The dataset [2] centers on DDoS Botnet attacks originating from IoT devices, providing detailed packet features. The primary goal is to contribute to DDoS attack prevention, aiming to enhance cybersecurity measures for safeguarding IoT ecosystems from malicious activities. The University of New Brunswick's dataset, created for DDoS analysis from 2018 [3], it captures DoS attacks and includes eighty columns, with emphasis on the Label column for identifying malicious packets. Notebook creators must balance the unbalanced files by date to enhance prediction accuracy.

3.2 Data Exploration

Prior to developing a model, it is critical to comprehend the features and organization of the IoT-23 information. Data scientists utilize exploratory data analysis (EDA) to examine and explore datasets, summarizing their primary attributes through the application of data visualization techniques. The techniques which are used in this study contains correlation, heatmaps, info, counts of certain values and many other.

3.3 Data Preprocessing

Before giving the data to model, data is preprocessed and is bought to the certain pattern using encoding, removing and changing the values of certain columns.

4. Machine Learning Models

4.1 Bagging Classifier:

Bagging, short for Bootstrap Aggregating, is an ensemble learning technique where multiple base models are independently trained in parallel on diverse subsets of the training data. These subsets are created through bootstrap sampling, randomly selecting data points with replacement. For the Bagging classifier, the ultimate prediction is determined by aggregating the predictions of all base models through majority voting. In regression, the final prediction involves averaging the predictions of all base models, referred to as bagging regression.

- ***Bootstrap Sampling:*** Randomly sample 'n' subsets of the original training data with replacement. Ensures diversity in training subsets, reducing overfitting risks and enhancing model accuracy.
- ***Base Model Training:*** Utilize multiple base models, each independently trained on different bootstrapped subsets.
  Base models, often termed "Weak learners," may include decision trees, support vector machines, or neural networks.
  Training is performed independently in parallel to enhance computational efficiency.
- ***Aggregation:*** Combine predictions from all base models to make predictions on unseen data.
  In the bagging classifier, employ majority voting to determine the predicted class label, ensuring robust predictions.
- ***Out-of-Bag (OOB) Evaluation:*** Exclude certain samples from the training subsets of specific base models during bootstrap sampling. Use "out-of-bag" samples to estimate model performance without requiring cross-validation.
- ***Final Prediction:*** Aggregate predictions from all base models to generate a final prediction for each instance in the dataset.

The Bagging Classifier offers several advantages, including enhanced predictive performance through the reduction of overfitting and increased accuracy by combining multiple base models. It demonstrates robustness by mitigating the impact of outliers and noise in the data, leading to a more stable model. The technique reduces variance by training each base model on different data subsets, and its parallelization capability makes it computationally efficient, especially for large datasets. Additionally, the Bagging Classifier is flexible and applicable to various machine learning algorithms.

4.2 Decision Tree

Choice The interpretability, simplicity, and ease of implementation of trees make them a popular choice. Recursively dividing the dataset according to characteristics is the first step in the decision-making process. This creates a structure like a tree, with each leaf node representing a choice.

Among Decision Trees' benefits are:

- ***Interpretability:*** Choice Trees offer a simple, intelligible method for making decisions. Every branch in the tree reflects a choice made in light of a particular feature, and every node in the tree represents a feature.
- ***Managing Non-linearity:*** Decision Trees are appropriate for datasets with complicated structures because they can manage non-linear interactions between features.
- ***Importance of characteristics:*** Decision Trees are designed to automatically rank characteristics according to their significance, which helps identify the key elements that influence the categorization.

The Decision Tree algorithm offers several advantages, including its simplicity, mirroring human decision-making processes and its applicability to decision-related problems. It encourages comprehensive consideration of possible outcomes and requires less data cleaning than alternative algorithms. However, Decision Trees come with drawbacks, such as their inherent complexity due to numerous layers, which can lead to challenges in interpretation. There is also a risk of overfitting, a concern addressed by the implementation of the Random Forest algorithm. Additionally, when dealing with numerous class labels, the computational complexity of Decision Trees may increase.

5. Methodology

5.1 Data Encoding Data from both dataset is preprocessed using "**Min-Max Scaler**" of Sklearn library. A fit transform calculates the mean and standard deviation and then scale the

data with respective to it. This processed Dataframe is then use further for Training.

5.2 Splitting data: Data is split in 80/20% using sklearn "test_train_split" into the respective Xtrain,Ytrain for training and xtest and ytest for testing.

5.3 Model Explanantion: Data is split in 80/20% using sklearn "test_train_split" into the respective Xtrain, Ytrain for training and xtest and ytest for testing.

6. Model Evaluation

6.1 Metrics

Metrics for evaluation are essential for comprehending how well any machine learning model performs. The model will be evaluated using a number of indicators.

- ***Accuracy:*** The percentage of cases accurately categorized relative to all instances.
- ***Precision:*** A measure of a model's ability to prevent false positives, expressed as the ratio of genuine positive predictions to all predicted positives.
- ***Recall (Sensitivity):*** The proportion of real positives to true positive forecasts, indicating how well the model identified all pertinent cases.
- ***F1 Score:*** A balanced metric between recall and accuracy, calculated as the harmonic mean of the two.
- ***ROC:*** The Receiver Operating Characteristic (ROC) curve is a graphical representation of a binary classification model's performance across various classification thresholds. It plots the trade-off between the true positive rate (sensitivity) and the false positive rate (1-specificity). The curve helps visualize how well the model discriminates between positive and negative instances. The area under the ROC curve (AUC-ROC) is a common metric, with a higher value indicating superior model performance.
- ***Confusion Matrix:*** A table that provides a thorough understanding of the performance of the model by showing true positive, true negative, false positive, and false negative values.
  The confusion matrix generated is as follows:

[[151681 46]
[20 57158]]

7. Visualization:

As described earlier decision tree uses the set of rules and constraints to predict the desire output. Using Sklearn a tree is constructed. Which is as follows.

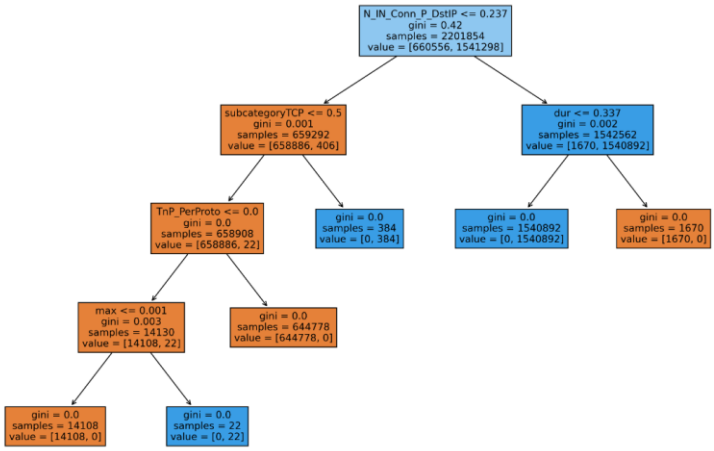

Figure 1: Decision Tree

Also, A ROC can be a better representation of analysis of the performance.

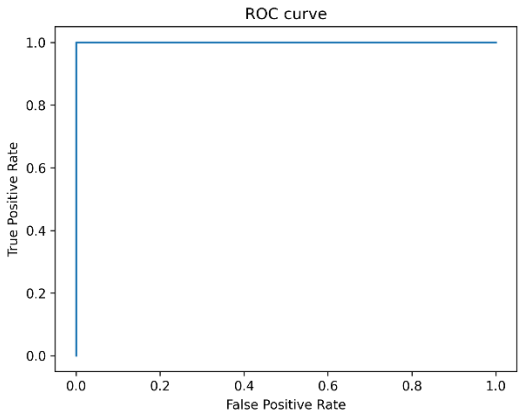

Figure 2: ROC

From the above graph it can be observed that ROC just boosted to 1 and stated over it.

8. Conclusion

8.1 Findings

As discussed in literature review many techniques, approaches and methods can be used for prediction of the attack but from the study and on the basis of certain evaluation metrices.

The Bagging Classifier offered enhanced predictive performance through the reduction of overfitting and increased accuracy up to 99.9% with addition of precision and recall. By using decision tree as its base model.

8.2 Limitations

The Basic limitation which is also the main one is that in this study the model is trained only on certain files and not all the files. From the observation and vast given dataset it can be formulated that after training it on whole dataset it may lead to the overfitting.

9. Future Work

9.1 Enhancements

The performance matrices had much increased value so there is no need of changing the model. But from the limitation data should be further trained and should be further use.

10.2 Real-World Application

Further study may entail implementing the models in real-world IoT contexts to test their feasibility. For practical implementation, it is imperative to assess their performance under a range of network situations and resolve model interpretability difficulties in security-critical applications.

This thorough investigation lays the groundwork for future developments in machine learning models for IoT device security.

11. References

[1] ·. H. V. L. H. S. K. P. T. K. S. Tong Anh Tuan, "Performance evaluation of Botnet DDoS attack detection using machine learning," 2019. [Online]. Available: https://link.springer.com/article/10.1007/s12065-019-00310-w.

[2] S. M, "DDoS Botnet Attack on IOT Devices," (2020). [Online]. Available: https://www.kaggle.com/siddharthm1698/ddos-botnet-attack-on-iot-devices.

[3] SOLARMAINFRAME, "IDS 2018 Intrusion CSVs (CSE-CIC-IDS2018)," 2021. [Online]. Available: https://www.kaggle.com/solarmainframe/ids-intrusion-csv.