SIADS 696 Milestone II Project Report

DDOS Attack Classification

Jalaj Rastogi (jrastogi@umich.edu), Manish Jakhi (mrjakhi@umich.edu)

Github Link:
https://github.com/mrjakhi/MADS_SIADS696_Team14_DDOS_detection.git

Introduction

What is a DDOS attack ?

"A distributed denial-of-service (DDoS) attack is a malicious attempt to disrupt the normal traffic of a targeted server, service or network by overwhelming the target or its surrounding infrastructure with a flood of Internet traffic."

Distributed denial of service (DDoS) attacks is a subclass of denial of service (DoS) attacks. Distributed Denial of Service attacks are a serious threat to organizations that provide internet-based services and applications. These attacks inhibit the server's ability to provide resources to genuine customers, they directly target servers resources such as bandwidth and buffer size to slow it down. A DDoS attack involves multiple connected online devices, collectively known as a botnet, which are used to overwhelm a target website with fake traffic. DDoS attacks can indeed have a devastating impact on internet-based services. These attacks are designed to overwhelm a target system or network with a flood of traffic, making it difficult or impossible for legitimate users to access the service or application. DDoS attacks can be launched using a variety of techniques, such as flooding the target with large volumes of traffic, exploiting vulnerabilities in network protocols, or using a botnet to coordinate the attack across multiple compromised devices.

The scale and sophistication of these attacks have increased over time, making them a significant threat to internet-based businesses and organizations. The impact of a DDoS attack can be severe, ranging from temporary disruption of services to long-term damage to a company's reputation and financial standing. In some cases, DDoS attacks have been used as a smokescreen for other malicious activities, such as data theft or system compromise.

To mitigate the risk of DDoS attacks, organizations need to implement a range of measures, including network segmentation, traffic filtering, and the use of DDoS protection services. It is also important to regularly test and update incident response plans to ensure that the organization is prepared to respond quickly and effectively to an attack. In this project our goal is to explore

different supervised and supervised machine learning techniques that could be deployed to identify a DDOS attack.

Difference between a DOS attack and a DDOS attack

The main difference between DoS and DDoS attacks is the number of sources of attack traffic used. DoS attacks typically use a single source of traffic to overwhelm a target system or network, while DDoS attacks use multiple sources, often in the form of a botnet, to coordinate the attack and increase its effectiveness.
While DoS attacks can be effective in certain circumstances, they are generally less powerful and easier to mitigate than DDoS attacks. DDoS attacks can generate much larger volumes of traffic and are often more difficult to detect and block, as the attack traffic is coming from multiple sources.

Related Works:

- A similar study has been conducted on DDOS attack classification where two Virtual machines (mininet and ryu-controller) running on Ubuntu are developed and configured to create a network topology where traffic can be adjusted and monitored and a labeled csv file was generated by monitoring the traffic. Attacks covered in this project are TCP, UDP, ICMP floods and Land attack.
- Another project is kaggle competition on CIC-IDS2018 AWS Dataset. This dataset does not only include DDOS attacks but also consists of attacks like Brute Force, SQL Injection etc.
- Lastly, just like the above project a similar DDOS attack dataset is available on kaggle (https://www.kaggle.com/datasets/jacobvs/ddos-attack-network-logs) with different features and. Attacks covered in this dataset are UDP Flood, Smurf, SIDDOS, HTTP Flood.

History of DDOS attacks ?

Distributed Denial of Service (DDoS) attacks have been around since the early days of the internet, and their history is closely tied to the evolution of network technologies and security.

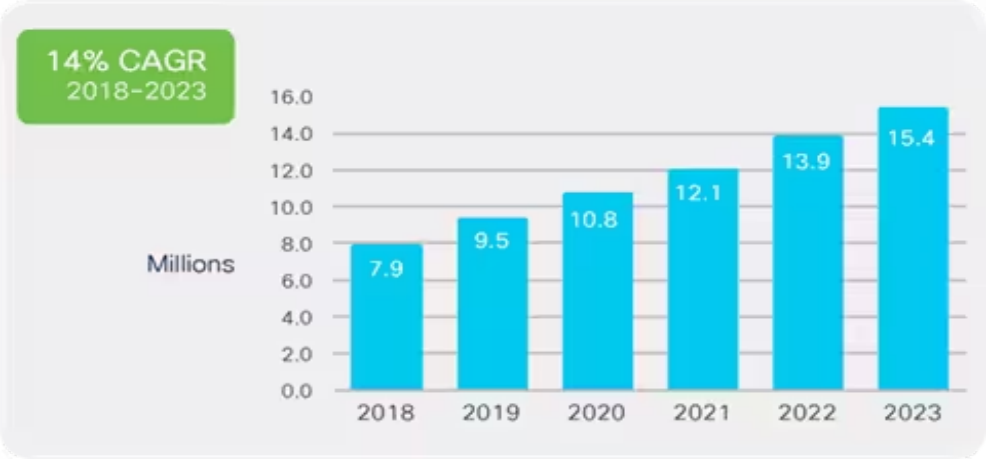


Figure 1. Cisco's analysis of DDoS total attack history and predictions.
Source: Cisco Annual Internet Report, 2018–2023

Five Most Famous DDoS Attacks

1. The Google Attack, 2020
   https://blog.google/threat-analysis-group/how-were-tackling-evolving-online-threats
2. The AWS DDoS Attack in 2020
   https://aws-shield-tlr.s3.amazonaws.com/2020-Q1_AWS_Shield_TLR.pdf
3. The Mirai Krebs and OVH DDoS Attacks in 2016
   https://www.a10networks.com/blog/inside-the-mirai-malware-that-powers-iot-botnets/
4. The Mirai Dyn DDoS Attack in 2016
   https://www.zdnet.com/home-and-office/networking/the-dyn-report-what-we-know-so-far-about-the-worlds-biggest-ddos-attack/
5. The GitHub Attack in 2018
   https://www.wired.com/story/github-ddos-memcached/

Cost of DDOS attack

The cost of a DDoS attack can vary widely depending on a number of factors, such as the duration and intensity of the attack, the target system or network, and the industry or sector being targeted. Some of the costs associated with a DDoS attack may include:

- Lost revenue: DDoS attacks can cause significant disruptions to online services and e-commerce platforms, leading to lost sales and revenue for affected businesses.
- Damage to reputation: DDoS attacks can also damage a company's reputation, particularly if the attack results in extended downtime or data loss.
- IT recovery costs: Businesses may incur significant costs in repairing or replacing damaged IT infrastructure and restoring service after a DDoS attack.
- Legal and regulatory costs: Depending on the industry and jurisdiction, businesses may face legal and regulatory compliance costs in the aftermath of a DDoS attack, including fines and penalties.
- Cost of DDoS protection: Businesses may need to invest in DDoS protection services or other measures to prevent future attacks, which can be costly.

Overall, the cost of a DDoS attack can be significant, both in terms of direct costs such as IT recovery and indirect costs such as lost revenue and damage to reputation. It is important for businesses to implement measures to mitigate the risk of DDoS attacks, such as regular security testing, redundancy and failover mechanisms, and DDoS protection services.

Information about dataset:

Our project involves using a pcapng file generated by Wireshark, a widely used tool for capturing and analyzing network packets, instead of pre-configured VMs and pre-generated datasets. We plan to extract features from the pcapng file using Wireshark and develop a machine learning model that can classify traffic as either a DDoS attack or benign traffic. To achieve our goals, we will use both supervised and unsupervised learning. The unsupervised learning task will involve classifying traffic as malicious or benign, while the supervised learning task will further classify DDoS traffic into 5 categories: Slow HTTP POST, Slow Read, Slow HTTP GET, Hulk, Goldeneye, along with benign traffic.

Data Source and Feature Engineering

There are two sources of data for this project. We used DDOS File which is a pcapng file and contains five different categories of ddos attack. The second one was downloaded from Normal Traffic and is also a pcapng file.

DDOS Data

The ddos traffic was generated in a live network where different nodes or devices were assigned an IP address and each node was configured to execute a particular type of ddos attack. The IP addresses of the devices and their configured attacks are as follows:

1. 192.168.1.64 was responsible for Slow HTTP GET
2. 192.168.1.65 was responsible for Slow Read
3. 192.168.1.66 was responsible for Slow Post
4. 192.168.1.67 was responsible for GoldenEye
5. 192.168.1.68 was responsible for HULK
6. 192.168.10.124 was the node responsible for receiving the traffic

The format for this data is a pcapng file which contains information about individual packets which flowed from one device to another. It contains 912986 packets and is about 562 MB in size.

Normal Traffic

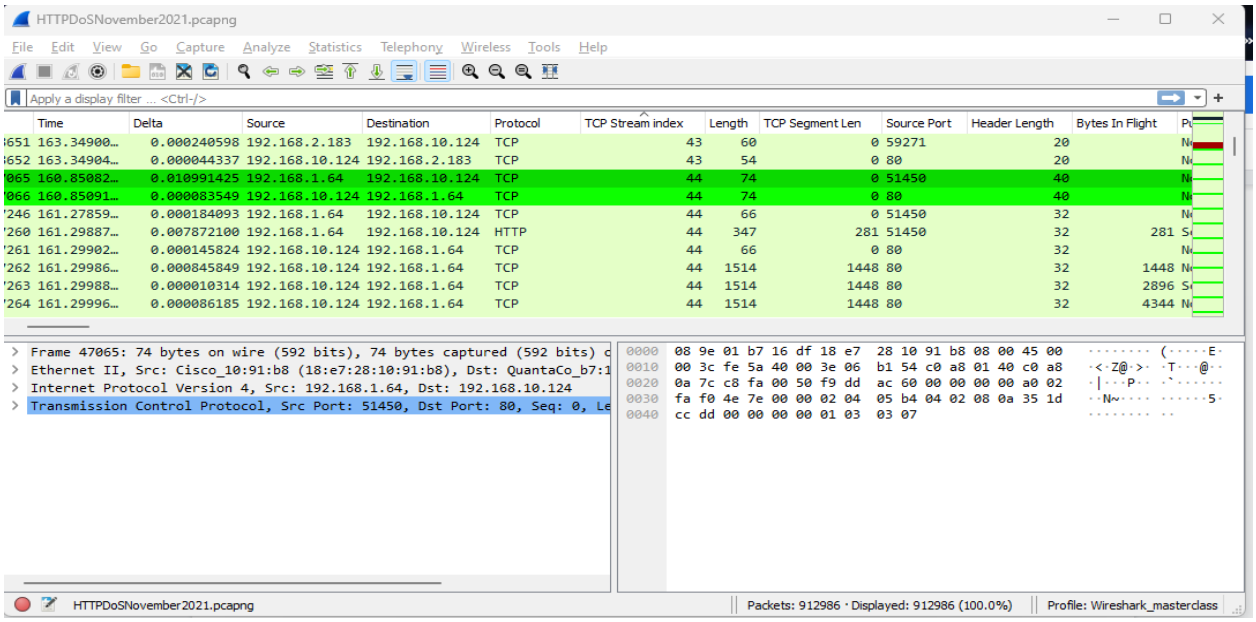
This is a capture of network traffic on a busy private network's access point to the Internet. Similar to DDOS attack file this is also in pcapng format, contains 791615 packets and is about 351 MB in size.

Feature Engineering

Feature Engineering was basically split between two different parts. The first part focuses on extracting features from wireshark and the second part focuses on using pandas to make the rough data extracted from wireshark into a csv file ready to be used by the models.

a. Wireshark Part

Whenever a pcapng file is opened in wireshark it shows a list of all the packets that were captured during traffic monitoring. The individual packets in wireshark look like the following screenshot.



The following steps were performed in wireshark:

1. Each IP address under the column “Source” performs a series of communications (sending and receiving packets) with the IP address under the destination column. In the screenshot above each row is an individual packet and a stream ID has been assigned to that packet by wireshark. We first grouped all these packets by the stream ID (TCP Stream Index column) and exported it to csv which had the following columns - "Address-A", "Port-A", "Address-B", "Port-B", "Packets", "Bytes", "StreamID", "TotalPackets", "PerctageFiltered", "Packets_A-to-B(TX_pkts)", "Bytes_A-to-B(TX_bytes)", "Packets_B-to-A(RX_pkts)", "Bytes_B-to-A(RX_bytes)", "Relative_Start", "Duration", "Bits/s A-to-B (TX_bits/s)", "Bits/s B-to-A (RX_bits/s)"

2. Once we had the above csv file we exported the table in the above screenshot which gave us a table of individual packets. The reason we had to do this because some of the features like “flags” were not being added to csv in step-1 when we tried grouping by stream ID in wireshark

The following steps were performed using pandas:

1. After the above steps we had two csv files one with packets grouped by stream ID and other containing individual packets. We counted the number of flags (PSH, RST, ACK, SYN, FIN) by mapping them to 1 if they were set and 0 if they were not set. After counting we grouped them by Stream ID and calculated their sum which was assigned to the “flags” column. For numeric columns like tot_kbps we aggregated them by their mean after grouping them by stream ID.
2. This was done separately for ddos file and normal attack file. Once we had the dataframes for both ddos and normal they were clubbed together as one single dataset. The numeric columns had to be scaled using StandardScaler for LogisticRegression and SVC

Below is the table list of features used

Column Name	Description	Data Type
Packets	Number of packets in a Stream ID	Numeric
Bytes	Number of Bytes in a Stream ID	Numeric
Packets_A-to-B(TX_pkts)	Number of packets sent from source address to the destination address in a Stream ID	Numeric
Bytes_A-to-B(TX_bytes)	Number of Bytes sent from source address to the destination address in a Stream ID	Numeric
Packets_B-to-A(RX_pkts)	Number of packets received by source address from the destination address in a Stream ID	Numeric
Bytes_B-to-A(RX_bytes)	Number of Bytes received by source address from the destination address in a Stream ID	Numeric
Bits/s A-to-B (TX_bits/s)	Number of bits per sec sent from source address to the destination address in a Stream ID	Numeric
Bits/s B-to-A (RX_bits/s)	Number of bits per sec received by source address from the destination address in a Stream ID	Numeric
tot_kbps	Total KBs sent and received in the Stream ID	Numeric
Pktrate	Packets per second sent and received in a Stream ID	Numeric
Flow Duration	The duration of the Stream ID	Numeric
Idle Timeout	Duration for which the Stream ID was idle	Numeric
Hard Timeout	It is the explicit limit for teh Stream ID after which the stream ID is closed.	Numeric
Pktrate nsec	Packet rate per nano sec	Numeric
Byte nsec	Bytes per nano sec	Numeric
flags	Sum of count of PSH, RST, ACK, SYN, FIN.	Numeric
CATEGORY (target column)	Category of stream : Benign Traffic, Slow HTTP GET, Slow Read, Slow Post, GoldenEye, Hulk	Categorical

Supervised

Motivation

The goal of our project is to develop a machine learning model that will help to predict whether a particular conversation between two ip addresses belongs to a category of DDOS attack or if it is benign. DDOS attacks have many categories but for this project we have only chosen 5 categories - Slow HTTP GET, Slow Read, Slow POST, Hulk, Goldeneye. A small description of each category is given in the appendix. As we have to predict a label and we have multiple labels it falls under the category of multinomial classification. We had many features that came from wireshark during data preprocessing and we had to drop some of the features such as source ip, destination ip, source port, destination port, flow id etc as they were not useful for the problem. The features used for supervised machine learning are provided in the table in the **Data Source and Feature Engineering section**.

Methods and Model Evaluation:

We have used three different models for our problem and then we selected the best performing model for further analysis:

1. Logistic Regression : This was treated as the baseline model and was implemented using sklearn.
2. SVC : This was selected as the second model and was also implemented using sklearn.
3. RandomForestClassifier : Tree based ensemble algorithm. Also implemented using sklearn

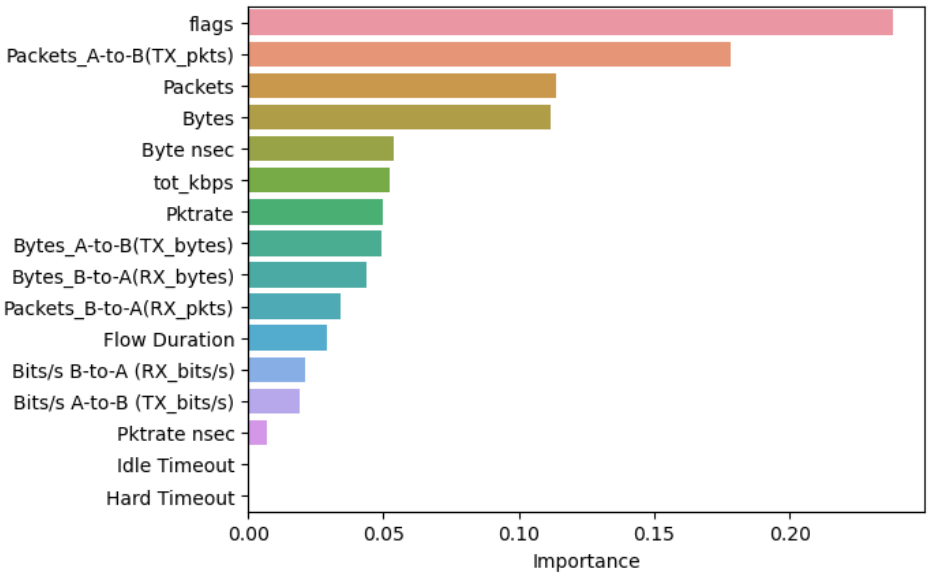
The evaluation metric - accuracy was used to judge the performance of the model. When we were implementing the model we had the problem of data imbalance. In order to solve this problem we used SMOTE and performed oversampling on the label “Slow Read” which comprised just 2% of the dataset. We experimented on the models’ performance both before oversampling and after oversampling. Without oversampling we were getting no predicted labels for “Slow Read” in classification reports for both Logistic Regression and SVC hence this made oversampling necessary for our experiment. Hence after oversampling we performed 5-fold cross validation to get the accuracy across different models for comparison

Model	Logistic Regression	SVC.svm model	RandomForestClassifier
Accuracy	0.5893 ± 0.0019	0.6366 ± 0.0031	0.9118 ± 0.0031

The RandomForestClassifer achieved the highest accuracy hence we chose to conduct further detailed analysis.

Feature Analysis

We used inbuilt feature importance in RandomForest to conduct feature analysis. We observed that flags feature had the highest weightage followed by Packets_A-to-B(TX_pkts), Packets and then finally Bytes. All these features are numerical columns. In order to further check their effect on model performance we also conducted an ablation analysis.



S No.	Ablated Features	Accuracy (5 fold cv)
1.	No features removed	0.9118 ± 0.0031
2.	flags	0.7433 ± 0.0079
3.	Packets_A-to-B(TX_pkts)	0.8687 ± 0.0058
4.	Packets, Bytes	0.9079 ± 0.0048

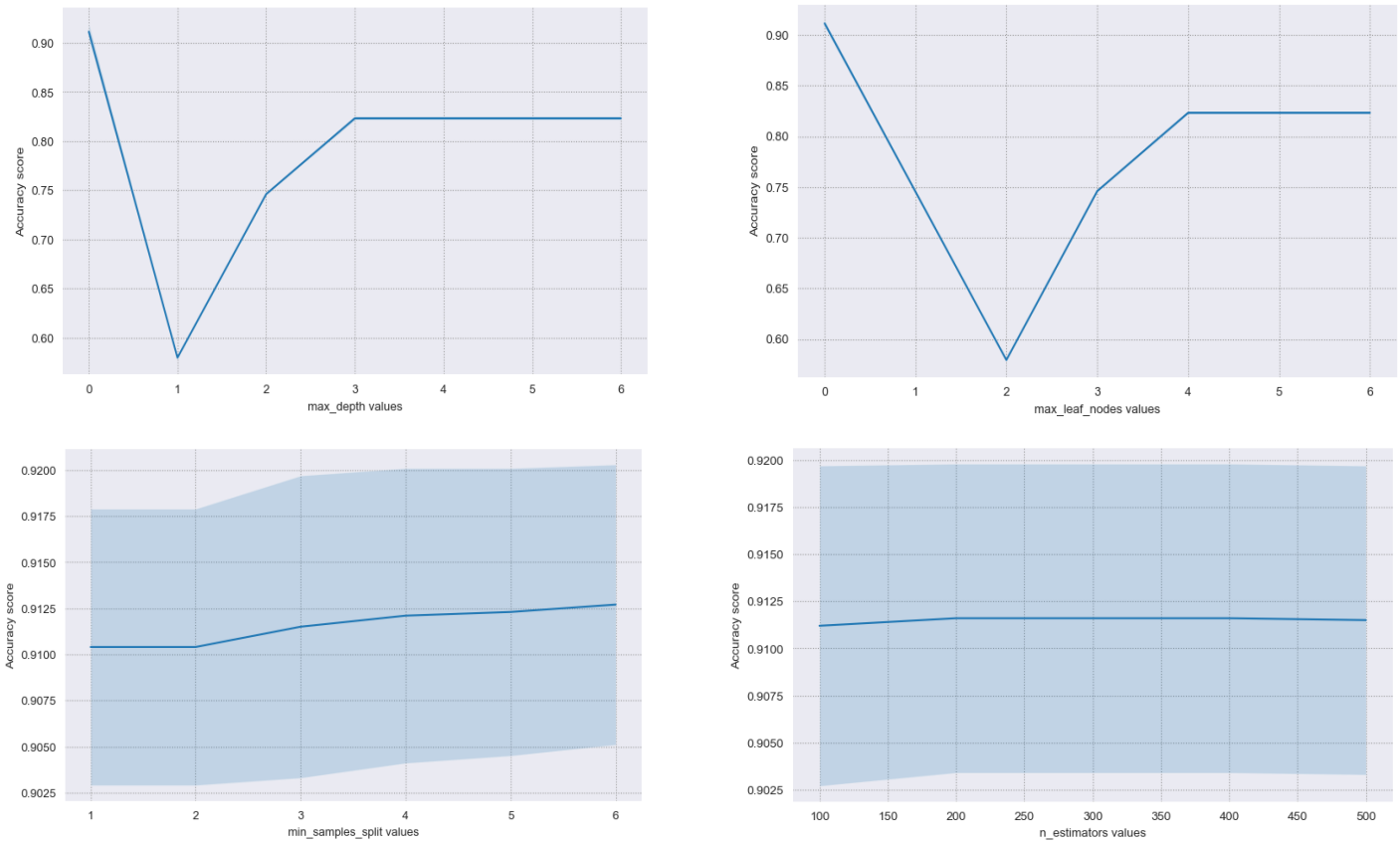
Key Findings:

1. The feature “flags” is a combination of several features which are extracted from wireshark. It is the sum of the count of the following flags - PSH, RST, ACK, SYN, FIN. The count of individual flags was extracted from wireshark and they were aggregated using pandas. An explanation of each abbreviation is in **Appendix C.** As soon as this column was removed we observed a decrease in accuracy of the model from 91% to 74%.
2. The second most important feature “Packets_A-to-B(TX_pkts)” is the number of packets that were transmitted from source to the destination. Generally when a conversation happens between two ip addresses, the packets send by source ip address are called forward packets or transmitted packets (TX packets) and the packets received by the source ip from the destination ip are called backward packets or received packets (RX packets). This feature too was calculated using wireshark after grouping the conversations by stream id. Hence the step of extracting this feature is important.

3. Packets and Bytes are features which were present by default in wireshark. No additional step was carried out in calculating these columns. Removing these columns did not have much effect on accuracy.

Sensitivity Analysis

We also performed sensitivity analysis for hyperparameters of our model by changing one hyperparameter while keeping others fixed. This was performed for “max_depth”, “max_leaf_nodes”, “min_samples_split”, “n_estimators”. We observed that the accuracy varied the most with “max_depth” and “max_leaf_nodes” whereas it was almost constant for “min_samples_split” and “n_estimators”.



The figures above from top left in clockwise direction are “max_depth vs Accuracy”, “max_leaf_nodes vs Accuracy”, “min_samples_split vs Accuracy” and “n_estimators vs Accuracy”.

Key Findings:

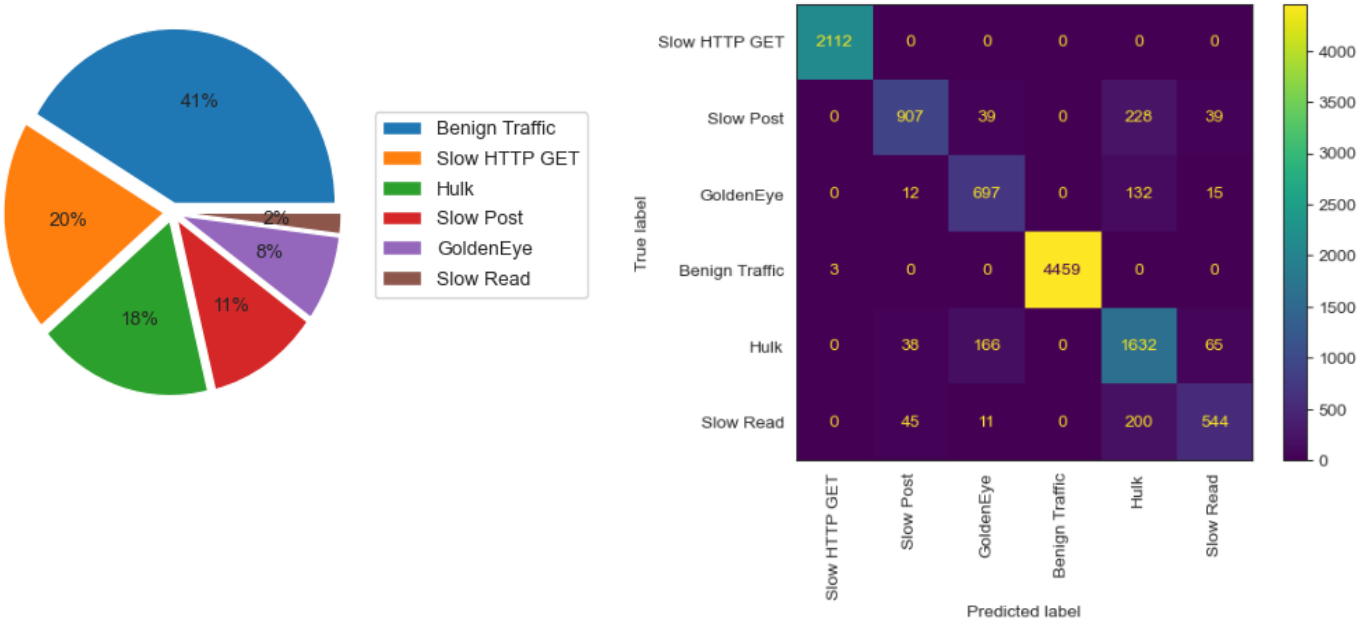
1. We observed that max_depth and max_leaf_nodes plots are almost similar to each other as after the 0 or “None” value there is a sharp fall at 1 for max_depth and 2 for max_leaf_nodes (didn't put 1 as value of 1 for max_leaf_nodes will throw exception). After the dip the accuracy becomes constant for both the hyperparameters. This can be a point for future

considerations as when we have more diverse data such as more categories of DDOS attacks as well as larger dataset such as having 1 million records or more.

2. The hyperparameters “min_samples_split” and “n_estimators” did not have much effect on the accuracy as we can see the graph line has quite a flat slope. Thus changing these hyperparameters will not bring about much changes in the model performance.

Error Analysis

In order to understand the misclassifications made by the model we tried to gain insight in order to find ways by which we could improve our model.



The first and most important issue was the imbalance dataset. As seen in the pie chart “Slow Read” DDOS attack only accounts for 2% of the data. While we tried to use SMOTE to oversample it and solve this problem it cannot be compared to real life examples. Moreover we can also see in the confusion matrix that “Hulk” DDOS attack has been incorrectly predicted for “Slow Read”, “Slow Post” and “Goldeneye”. The following are the three example where Hulk DDOS attack was confused with other three mentioned attacks

Packets	Bytes	Packets_A-to-B(TX_pkts)	Bytes_A-to-B(TX_bytes)	Packets_B-to-A(RX_pkts)	Bytes_B-to-A(RX_bytes)	Bits/s A-to-B (TX_bits/s)	Bits/s B-to-A (RX_bits/s)	tot_kbps	Pktrate	Flow Duration	Idle Timeout	Hard Timeout	Pktrate nsec	Byte nsec	flags
0	0	0	3733631	0	1972979	266	143.0	0.0	0.0	13	20	100	0.0	0.0	34.0

```
test value original : Slow Post
predicted value : Hulk
```

Packets	Bytes	Packets_A-to-B(TX_pkts)	Bytes_A-to-B(TX_bytes)	Packets_B-to-A(RX_pkts)	Bytes_B-to-A(RX_bytes)	Bits/s A-to-B (TX_bits/s)	Bits/s B-to-A (RX_bits/s)	tot_kbps	Pktrate	Flow Duration	Idle Timeout	Hard Timeout	Pktrate nsec	Byte nsec	flags
0	0	0	4253	0	215596881	0	6138.0	0.0	0.0	6	20	100	0.0	0.0	22.0

```
test value original : Slow Read
predicted value : Hulk
```

Packets	Bytes	Packets_A-to-B(TX_pkts)	Bytes_A-to-B(TX_bytes)	Packets_B-to-A(RX_pkts)	Bytes_B-to-A(RX_bytes)	Bits/s A-to-B (TX_bits/s)	Bits/s B-to-A (RX_bits/s)	tot_kbps	Pktrate	Flow Duration	Idle Timeout	Hard Timeout	Pktrate nsec	Byte nsec	flags
0	0	0	69095	0	135525208	0	0.0	0.0	0.0	0	20	100	0.0	0.0	33.0

```
test value original : GoldenEye
predicted value : Hulk
```

In the feature importance analysis earlier we found out that “flags” is the most important feature. We observed that the range for “flags” for Hulk (28±4) is quite close to that of “Slow Read” (32±8), Slow Post (16±10), and GoldenEye(29±7).

The suggestions for the future are as follows:

1. In the flags column we only included the count of the following flags - PSH, RST, ACK, SYN, FIN. There are more flags which could be explored and experimented upon.
2. Extraction of additional features from wireshark. Due to time constraint and wirehsark being a complicated tool we were not able to get fully accustomed to it. There might be more features which could be extracted from wireshark which could give better insights about the packets.
3. Duration of the traffic could be increased and as a result the dataset size will also increase. If we want a bigger dataset we might have to include other categories of DDOS attacks like UDP flood, cookie MHDDOS, DNS Request Flood etc. This will also include addition of more features into the dataset.

Unsupervised

Motivation

Our goal here is to determine whether supervised learning methods can be used to correctly classify DDOS attacks into one of these 2 categories : “Malignant” or “Benign”. The motivation for using unsupervised learning techniques such as k-means clustering, t-SNE and principal component analysis (PCA) for DDoS detection lies in their ability to detect patterns and anomalies in large volumes of data without the need for labeled training data.
DDoS attacks can generate a massive amount of traffic, making it difficult to identify and differentiate malicious traffic from legitimate traffic. Unsupervised learning algorithms such as k-means clustering can group together similar traffic patterns based on their features or attributes, allowing for the identification of abnormal traffic that does not fit the expected patterns.

In addition, PCA can be used to reduce the dimensionality of the data, simplifying the clustering process and making it more efficient. By reducing the number of features or variables used in the clustering process, PCA can help to identify the most important features and reduce noise or irrelevant data.
Overall, the use of unsupervised learning techniques such as k-means clustering along with t-SNE and PCA can help to detect and mitigate DDoS attacks in real-time, allowing for faster and more accurate response times to potential threats.

Methods

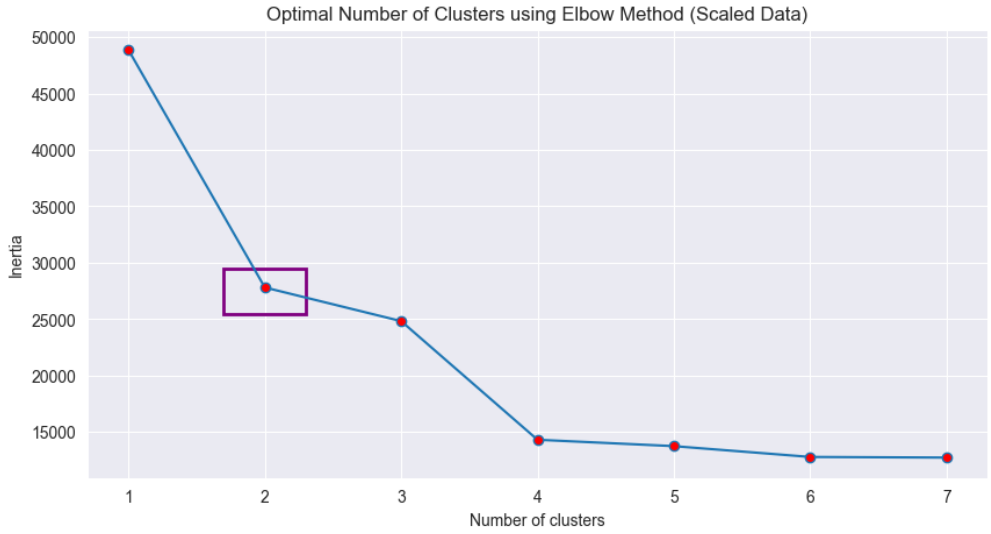
For an unsupervised learning approach we have used various initialization strategies for k-means algorithms. The initialization step of K-means is important because it can greatly affect the runtime and quality of the results. Below are the different method we used

1. Random Initialization (baseline): This is the simplest initialization strategy, where the initial centroids are randomly selected from the data.
2. Mini-batch K-means Initialization: This method is a variant of K-means that uses a random subset of the data to initialize the centroids, making it faster than the traditional K-means algorithm.
3. Density-Based Initialization: This method uses density-based clustering techniques to identify areas of high density in the data and select the initial centroids from those areas. This approach can be effective for datasets with unevenly distributed data points.
4. Init based on PCA projection: This method uses the components of the PCA to initialize KMeans. This method is deterministic.

Feature Tuning:

Optimal Number of clusters evaluation - using Elbow method

To determine the optimal number of clusters, we have to select the value of k at the “elbow” i.e. the point after which the distortion/inertia start decreasing in a linear fashion. It seems 2 or 4 clusters would be best. For the sake of simplicity we’ll select 2.



Model Evaluation:

For model evaluation we used Precision and Recall as our metric that we want to evaluate our model performance with. BisectingKMeans performed better in terms of precision and recall scores compared to other clustering algorithms such as random k-means, MiniBatch KMeans, and PCA-based KMeans. From here we concluded that BisectingKMeans is the most effective algorithm for clustering for this specific dataset.

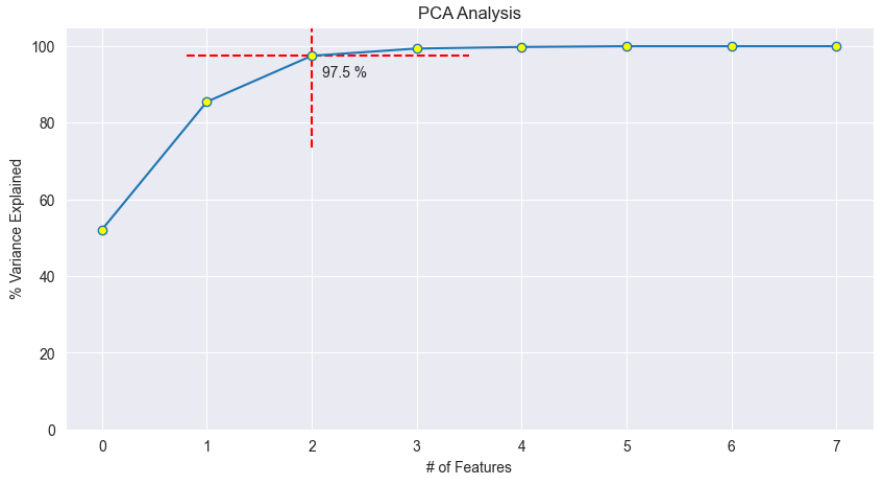
init	time	inertia	Precision	Recall	F1	Homo	compl	v-meas	ARI	AMI
random	0.061s	281132	0.533	0.284	0.371	0.004	0.004	0.004	0.013	0.004
MiniBatch	0.100s	292258	0.568	0.076	0.134	0.000	0.000	0.000	0.002	0.000
BisectingKMeans	0.030s	275897	0.633	1.000	0.776	0.106	0.266	0.151	0.085	0.151
PCA-based	0.020s	281537	0.476	0.254	0.331	0.016	0.018	0.017	0.036	0.017

To go even a step further we used the silhouette score, it is a metric used to evaluate the quality of clustering results. It takes into account both the cohesion of data points within a cluster and the separation between clusters. The silhouette score ranges from -1 to 1, where:

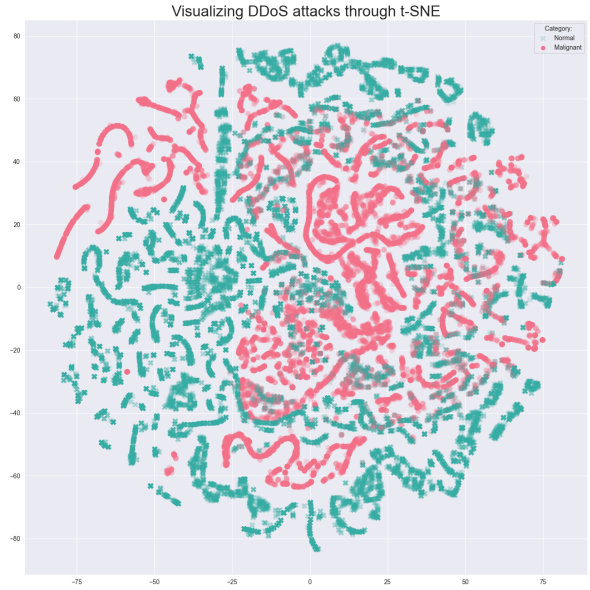
- A score of 1 indicates that clusters are well separated and distinct from each other, with cohesive and tightly packed data points within each cluster.
- A score of 0 indicates that clusters are overlapping or that the distance between them is not significant.
- A score of -1 indicates that clusters are poorly separated, with many data points misclassified or assigned to the wrong cluster.

In general, a higher silhouette score indicates better clustering results, with well-defined and well-separated clusters. However, the ideal score may vary depending on the specific context and data set being analyzed. It is important to use other evaluation metrics and domain knowledge to interpret and validate clustering results. Determining the silhouette score with number of clusters = 2, we got a KMeans Silhouette Score: 0.47393343479882527

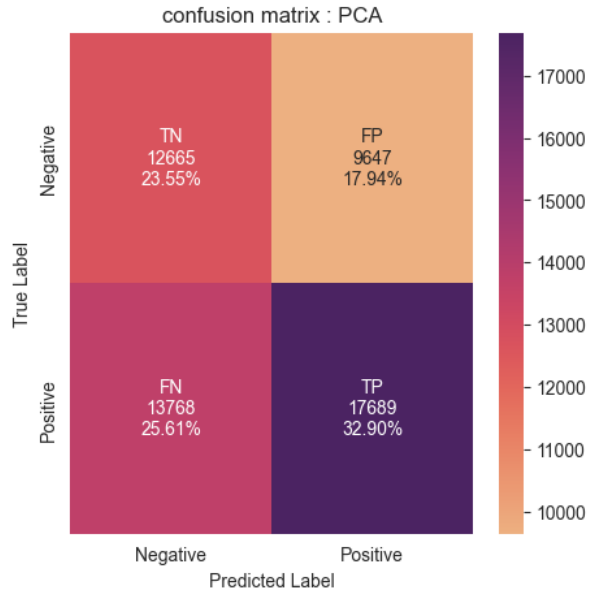
As a next step we looked into improving the model performance by feature reduction techniques like PCA and t-SNE. For this the first thing we did was to determine, number of optimal principal components needed. By examining the amount of variance each principal component encompasses we can see that the first 2 principal components explain roughly 97.5% of the variance.

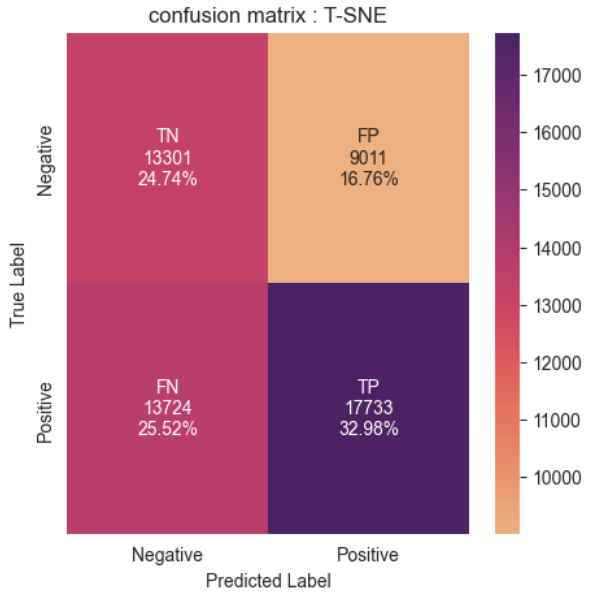


We then applied BisectingKMeans to the 2 feature reduced principal component dataset and recalculated Silhouette. The Score improved from 0.47 to 0.53 after applying dimensionality reduction through the Kmean.
We did the similar application but with 2 feature component t-SNE dataset. The SilhouetteScore decreased from 0.47 to 0.3323664367198944 in this case.
We visualized the cluster formed through PCA and t-SNE



Plotted the confusion matrix to determine the performance of both the classification models





The confusion matrix above compares the predicted and actual values of the dataset and informats us about the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) in the classification. Here the true positive count is more than the other values in the confusion matrix, it means that the model has correctly identified a larger number of positive samples as positive. This is a good sign as it indicates that the model is accurately

identifying the positive class and is performing well in terms of precision, recall, and overall accuracy. However, it is important to look at the other values in the confusion matrix as well, the model still has a significant false positive rate which indicates that the model is incorrectly classifying some negative samples as positive, which can be problematic in this application where we are trying to determine whether the message packed received is an attack or not. In general, a high true positive count is a positive indicator of a model's performance, but when evaluated in conjunction with other metrics and factors the effectiveness of the model is questionable.

Discussions:

For this project the biggest challenge is understanding the external tool we used, which is wireshark. Since we used this tool for the first time we had to go through various tutorials and also study about various network terminologies in order to better understand our dataset. There are still a lot of features that could be explored and extracted using wireshark but we were not able to due to time constraints. Overall getting the data ready from a pcapng file to csv file which can be used by the models took around 4-5 weeks.
The second challenge was selecting features from the dataset. There were quite a lot of features when we were combining the two csv files generated from wireshark. Finding out which features were relevant for our experiment, which features were collinear was another time taking effort. It took a lot of time to understand what each feature represents and removing features which had different names but were duplicates of other features.
While doing feature importance analysis we were surprised to see “flags” as being the most important feature as compared to “Bytes” and “Packets”. The reason why we thought “Bytes” and “Packets” would be first is because when a SOC analyst tries to look for DDOS attack patterns in wireshark, he or she looks at the number and the size of request. The larger the size and more the number of requests means it might be a case of ddos attack. Moreover in the “flags” column we did not include all the flags and aggregated the count of five most common ones. Hence a more detailed and thorough experiment will be required to see how the model performs when other types of flags (for eg URG, CWE, ECE) are also taken into consideration.

Ethics discussion:

- Data privacy is a crucial ethical consideration when working with pcapng files or any other type of network traffic data. The information contained in these files can potentially reveal sensitive information about an organization's network, including vulnerabilities that could be exploited by attackers.
- One way to address this issue is to anonymize the IP addresses in the dataset before using it to train a model. This can be done by replacing the actual IP addresses with randomly generated identifiers, while still preserving the network traffic patterns and other relevant features of the data.
- Another important step is to ensure that any models trained on the data are used only for legitimate purposes, such as network security or performance optimization. It is important to

have clear policies in place for handling and securing the data, and to ensure that only authorized personnel have access to it.
- In addition, it is important to be transparent with users about how their data is being collected and used, and to obtain their consent whenever possible. This can help to build trust and promote ethical data practices.
- Overall, protecting data privacy is a critical consideration when working with network traffic data, and it is important to take steps to ensure that sensitive information is safeguarded and used only for legitimate purposes.

Statement of Work

- jalajrastogi4@gmail.com to get data access and raw data available
- jalajrastogi4@gmail.com / Manish Jakhi to work together to clean the dataset
- jalajrastogi4@gmail.com / Manish Jakhi to work together to define feature sets for both supervised and unsupervised methods
- jalajrastogi4@gmail.com to work on training models for different supervised methods.
- Manish Jakhi to work on training models for different unsupervised methods.
- jalajrastogi4@gmail.com / Manish Jakhi to collaborate on final model selection based on performance.
- jalajrastogi4@gmail.com / Manish Jakhi to document their work in the report individually first, followed up with collective reviews for completeness, documentation of conclusions and next steps.

References

(2020, March 9). *Cisco Annual Internet Report (2018–2023) White Paper*. Cisco.com. Retrieved February 26, 2023, from https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

(2022, January 21). *Five Most Famous DDoS Attacks and Then Some*. Www.A10networks.com. Retrieved February 26, 2023, from https://www.a10networks.com/blog/5-most-famous-ddos-attacks/

https://kb.mazebolt.com/kbe_taxonomy/http-based-ddos-attacks/

https://www.netscout.com/what-is-ddos/

Appendix A

Abbreviations:

Flags	Description
PSH	It stands for Push flag. It determines when a push flag is set or not
SYN	It is a flag that is set when a source address wants to start a TCP handshake with a destination address. The source address will send a packet with SYN flag set if it wants to establish a connection with destination address
ACK	This stands for Acknowledgement. When a SYN packet is sent from source address to destination address, the destination address sends a SYN-ACK packet to the source asking the source address to confirm. In response to this packet the source then sends ACK packet which confirm the establishment of the connection. So the sequence for connection is SYN packet from source to destination, SYN-ACK from destination to source and finally ACK packet from source to destination
RST	It stands for reset. It states that the source should delete the connection as the TCP segment does not meet the criteria for connection. Thus based on this fag the source resets or deletes the connection with destination
FIN	This flag is sent when the connection between the source and destination is broken. To end the connection gracefully between the source and destination this flag is set.

DDOS attack type information:

HULK : It is an attack similar to HTTP flood and is designed to overwhelm web servers' resources by continuously requesting single or multiple URL's from many source attacking machines. HULK flood differs from most available DDoS attack tools which produced predictable repeated patterns that could easily be mitigated. The principle behind the HULK flood is that a unique pattern is generated at each and every request, with the intention of increasing the load on the servers as well as evading any intrusion detection and prevention systems.

GoldenEye : GoldenEye is similar to HTTP flood and is a DDOS attack designed to overwhelm web servers' resources by continuously requesting single or multiple URLs from many source attacking machines. GoldenEye changes generated requests dynamically – it randomizes user agents, referrers and almost all of the various parameters used. GoldenEye attempts to keep the connection alive and also adds a suffix to the end of URLs which will allow the request to bypass many CDN systems (Also known as "No Cache"). When the servers' limits of concurrent connections are reached, the server can no longer respond to legitimate requests from other users.

Slow POST : In a Slow Post DDoS attack, the attacker sends legitimate HTTP POST headers to a Web server. In these headers, the sizes of the message body that will follow are correctly specified. However, the message body is sent at a painfully low speed. These speeds may be as slow as one byte every two minutes.

Slow Read : A slow read DDoS attack involves an attacker sending an appropriate HTTP request to a server, but then reading the response at a very slow speed, if at all. By reading the response slowly – sometimes as slow as one byte at a time – the attacker prevents the server from incurring an idle connection timeout. Since the attacker sends a Zero window to the server, the server assumes the client is actually reading the data and therefore keeps the connection open. This has the cumulative effect of consuming server resources, thus preventing legitimate requests from going through.

Slow HTTP GET : Slow HTTP GET attacks are denial-of-service (DoS) attacks in which the attacker sends HTTP requests piece by piece at a slow pace to a web server. If an HTTP request is not complete, or if the transfer rate is very low, the server keeps its resources busy waiting for the rest of the data