DDoS Detection Using Multi-Layer Perceptron

Submitted By:
Somya Saraswat (15MI409)
Aarjav Parashar (15MI411)

Security in any sense has three main pillars Confidentiality, Integrity & Availability. Where in Information Security Confidentiality means that information be only viewable/usable to those who are authorized. Integrity means that any changes in information be easily detectable and attributable onto the person who changed it ie. modifications should not be undetectable. The last one, Availability is what this project addresses. Availability means that a resource be always available to the authorized and verified users eg. An online shopping website should perform all its core functions properly whenever asked by an online visitor to do so.

Availability is extremely critical for a proper functioning internet. This is why organizations spend huge sums of money making sure their online services are always available as even a few minutes of down time can cost millions. This is the reason why a large number of cyber attacks involve “Availability”.

DoS or Denial of service attack is a general type of attack which can be conducted on any OSI network layer and can have consequences ranging from nothing to an entire system failure, it depends on how secured the target is and how well thought the attack is. DoS attacks are different from DDoS attacks in the sense that DoS attacks have a single point of initiation ie a single computer or network node. But DDoS attack can have billions of origination points depending on how many network nodes are participating in attack. This makes it extremely dangerous as with a large enough number of participants the attacker can theoretically bring down the entire internet infrastructure.

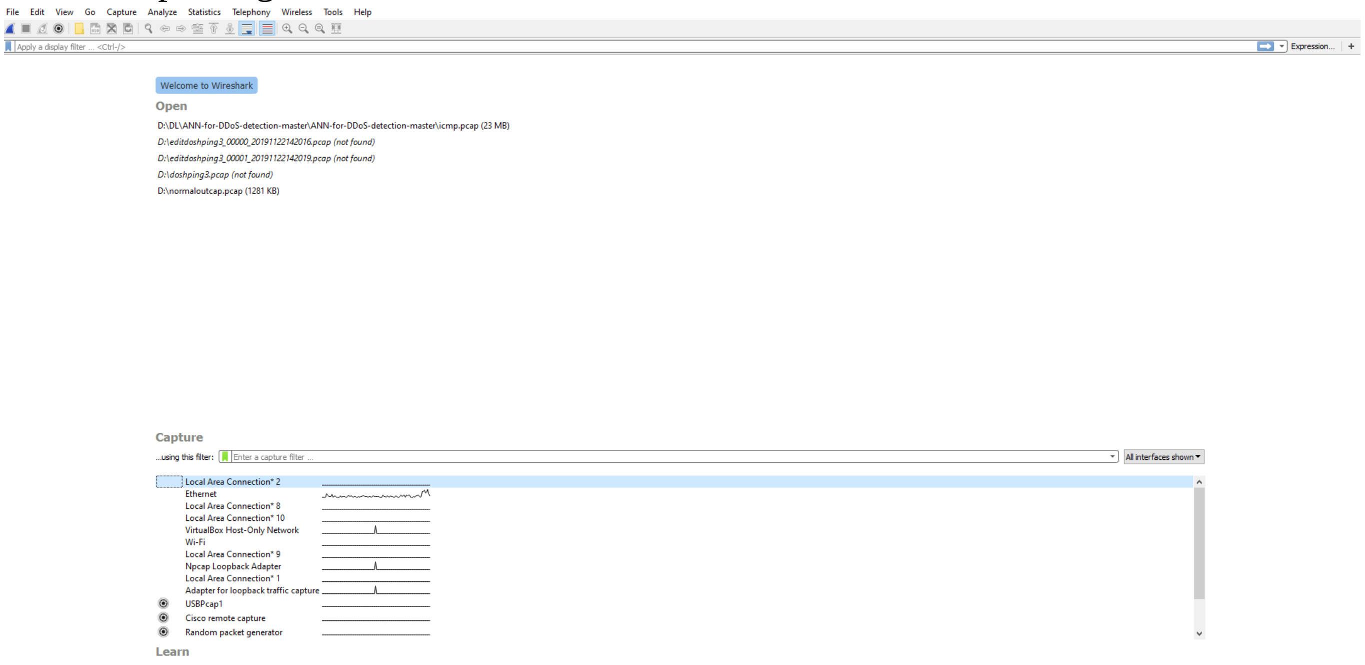
Working:

The entire project works in several stages.

1. Packet Capturing: In this stage you need to use wireshark to capture network traffic from the interface of your choice.
2. Dataset Generation: In this phase you'll have to run the included python script which converts pcap files parsing them into csv type files which will then be used in the Model.
3. Model Training stage: In this stage you Load up the generated training dataset and using a mixture of test and train samples you can train the model and later save it for further use or you can load a previously saved model and use it on a new test dataset.

Data Generation and Collection:

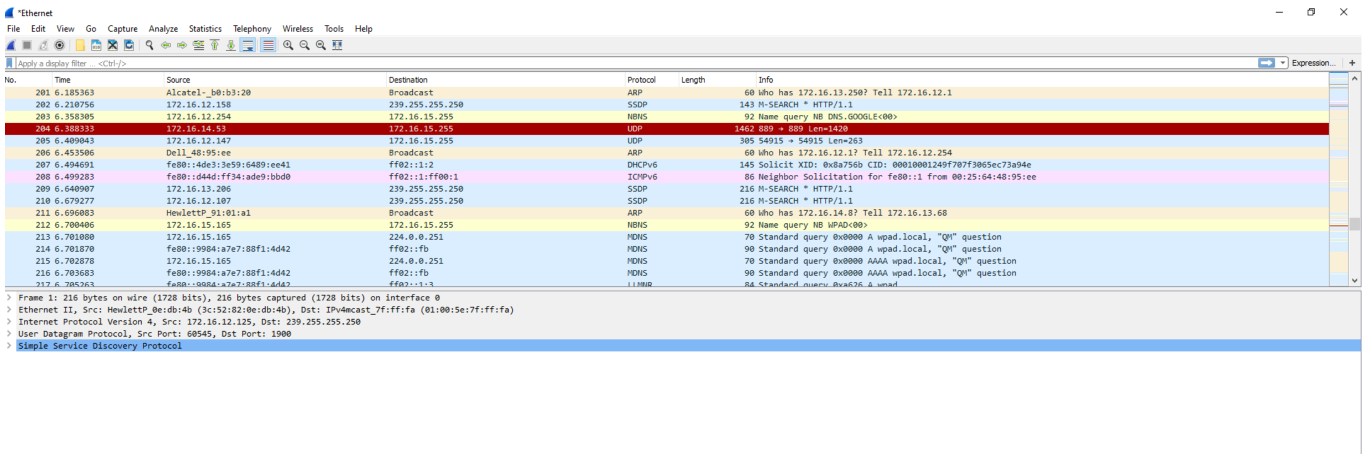
Packet Capturing:


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
Welcome to Wireshark
Open
Capture
...using this filter:
Enter a capture filter ...
All interfaces shown
Learn

D:\DL\ANN-for-DDoS-detection-master\ANN-for-DDoS-detection-master\icmp.pcap (23 MB)
D:\editdoshping3_00000_20191122142016.pcap (not found)
D:\editdoshping3_00001_20191122142019.pcap (not found)
D:\doshping3.pcap (not found)
D:\normaloutcap.pcap (1281 KB)

Local Area Connection* 2
Ethernet
Local Area Connection* 8
Local Area Connection* 10
VirtualBox Host-Only Network
Wi-Fi
Local Area Connection* 9
Npcap Loopback Adapter
Local Area Connection* 1
Adapter for loopback traffic capture
USBPcap1
Cisco remote capture
Random packet generator

Here, Double click on the network Interface you wish to capture packets on.


*Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

No.	Time	Source	Destination	Protocol	Length	Info
201	6.185363	Alcatel-_b0:b3:20	Broadcast	ARP	60	Who has 172.16.13.250? Tell 172.16.12.1
202	6.210756	172.16.12.158	239.255.255.250	SSDP	143	M-SEARCH * HTTP/1.1
203	6.358305	172.16.12.254	172.16.15.255	NBNS	92	Name query NB DNS.GOOGLE<00>
204	6.388333	172.16.14.53	172.16.15.255	UDP	1462	889 → 889 Len=1420
205	6.409043	172.16.12.147	172.16.15.255	UDP	305	54915 → 54915 Len=263
206	6.453506	Dell_48:95:ee	Broadcast	ARP	60	Who has 172.16.12.1? Tell 172.16.12.254
207	6.494691	fe80::4de3:3e59:6489:ee41	ff02::1:2	DHCPv6	145	Solicit XID: 0x8a756b CID: 00010001249f707f3065ec73a94e
208	6.499283	fe80::d44d:ff34:ade9:bbd0	ff02::1:ff00:1	ICMPv6	86	Neighbor Solicitation for fe80::1 from 00:25:64:48:95:ee
209	6.640907	172.16.13.206	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
210	6.679277	172.16.12.107	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
211	6.696083	HewlettP_91:01:a1	Broadcast	ARP	60	Who has 172.16.14.8? Tell 172.16.13.68
212	6.700406	172.16.15.165	172.16.15.255	NBNS	92	Name query NB WPAD<00>
213	6.701080	172.16.15.165	224.0.0.251	MDNS	70	Standard query 0x0000 A wpad.local, "QM" question
214	6.701870	fe80::9984:a7e7:88f1:4d42	ff02::fb	MDNS	90	Standard query 0x0000 A wpad.local, "QM" question
215	6.702878	172.16.15.165	224.0.0.251	MDNS	70	Standard query 0x0000 AAAA wpad.local, "QM" question
216	6.703683	fe80::9984:a7e7:88f1:4d42	ff02::fb	MDNS	90	Standard query 0x0000 AAAA wpad.local, "QM" question
217	6.705263	fe80::9984:a7e7:88f1:4d42	ff02::1:3	LLMNR	84	Standard query 0xa626 A wpad

```
> Frame 1: 216 bytes on wire (1728 bits), 216 bytes captured (1728 bits) on interface 0
> Ethernet II, Src: HewlettP_0e:db:4b (3c:52:82:0e:db:4b), Dst: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
> Internet Protocol Version 4, Src: 172.16.12.125, Dst: 239.255.255.250
> User Datagram Protocol, Src Port: 60545, Dst Port: 1900
> Simple Service Discovery Protocol
```

Then click on blue Shark fin and capture will start. Clicking on the red square button will stop the collection and on trying to exit it will ask you if you want to save the capture. Save it.
Then run the python script named pcaptocsv.py in the same directory as the pcap capture file, provide the script with the proper name when asked and wait for it to complete the process.
You will then have a .csv file as the output which can then be used as dataset.

Layer:

We used Multi-Layer Perceptron for this purpose with 2 hidden layers each containing 15 neurons each. A multilayer perceptron (MLP) is a class of feedforward artificial neural network (ANN). An MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

Activation = logistic
the logistic sigmoid function, returns f(x) = (1 / (1 + exp(-x)).

Solver = adam
'adam' refers to a stochastic gradient-based optimizer.

Shuffle = True
This shuffles the dataset entries over iterations to prevent the model from overfitting.

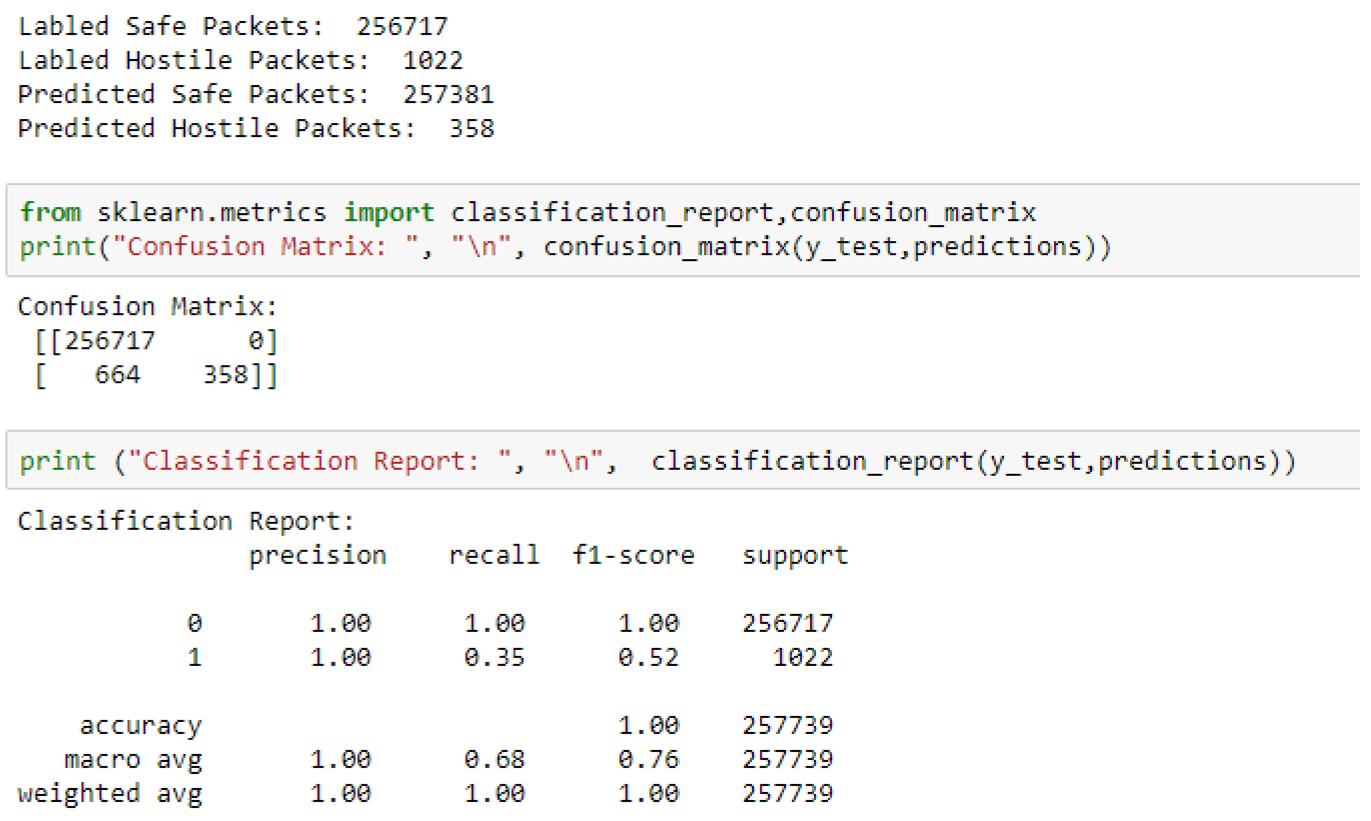
Learning_rate_init = 0.0001
The initial learning rate used. It controls the step-size in updating the weights.

early_stopping = True
Whether to use early stopping to terminate training when validation score is not improving.

tol = 0.0001

Tolerance for the optimization. When the loss or score is not improving by at least tol for 10 consecutive iterations, unless learning_rate is set to ‘adaptive’, convergence is considered to be reached and training stops.

Results:



```
Labled Safe Packets:  256717
Labled Hostile Packets:  1022
Predicted Safe Packets:  257381
Predicted Hostile Packets:  358
```

```
from sklearn.metrics import classification_report,confusion_matrix
print("Confusion Matrix: ", "\n", confusion_matrix(y_test,predictions))
```

```
Confusion Matrix: 
 [[256717      0]
 [   664    358]]
```

```
print ("Classification Report: ", "\n",  classification_report(y_test,predictions))
```

```
Classification Report: 
               precision    recall  f1-score   support

           0       1.00      1.00      1.00    256717
           1       1.00      0.35      0.52      1022

    accuracy                           1.00    257739
   macro avg       1.00      0.68      0.76    257739
weighted avg       1.00      1.00      1.00    257739
```

The model gives a Validation score of around 97-99% with small datasets having similar types of packets.

Requirements:

Following Python Modules are required:

- pyshark
- time
- datetime
- csv
- pickle
- pandas
- timeit
- sklearn

Following softwares are required:

- Wireshark
- tshark(If running pcaptocsv.py on linux)