Advances In Intrusion Detection System Using Different Classifiers

A Project Report

CS658A: Topics in Malware Analysis and Intrusion Detection

Under the guidance of

Prof. Sandeep Kumar Shukla

by Team Pegasus

Akanksha Singh (21111005) akankshas21@iitk.ac.in

Amar Raja (21111009) amard21@iitk.ac.in

Ankit Raja (21111012) ankitr21@iitk.ac.in

Ayush Sahni (21111019) sayush21@iitk.ac.in

Ayush Singh (21111020) ayushs21@iitk.ac.in

INDIAN INSTITUTE OF TECHNOLOGY, KANPUR
Department of Computer Science and Engineering

Abstract

Network Security is to protect computer network against hacking, misuse, unauthorized changes to the system and securing a computer network infrastructure. Intrusion Detection system (IDS) has become a prerequisite in computer networks. Based on the intrusion detection method, it is classified as Signature based and Anomaly based IDS.
Signature based IDS is that they operate by searching for a known identity or signature. An Anomaly based IDS is a system for detecting computer intrusions by monitoring system activity and classifying it as either normal or anomalous. Major drawback of anomaly based IDS/IPS is that it generates more false positive alarm.

Our proposed model is to implement the architecture of multimodel based Anomaly IDS with Neural Network (NN), Long short-term memory (LSTM) and Random Forest. We have integrated NN with Hidden Markov Model to improve our model. We have also tested our model by performing realtime attack on our model.

1 Motivation and Aim of the Project

An Anomaly based Intrusion Detection/Prevention System is a system for detecting computer intrusions by monitoring system activity and classifying it as either normal or anomalous. It consists of a statistical model of a normal network traffic which consists of the bandwidth used, the protocols defined for the traffic, the ports and devices which are part of the network. It regularly monitors the network traffic and compares it with the statistical model. In case of any anomaly or discrepancy, the administrator is alerted. An advantage of this system is they can detect new and unique attacks. But Anomaly based IDS have a drawback that it gives more false positive alarm.

To overcome the limitations of anomaly based intrusion detection, a classifier based on an intelligent hidden Markov model, is designed to differentiate the actual attacks from faulty ones. With this project, we intend to built a intrusion detection system that is more robust, accurate, reliable and gives less false positive alarms. So to achieve such an IDS we integrated three Intrusion detection system model together and then used voting system to predict intrusions in the system.

2 Proposed Idea

We proposes a multimodel based Anomaly IDS with Neural Network (NN), Long short-term memory (LSTM) and Random Forest. We integrated Neural Network with Hidden Markov Model to improve our Neural Network model. Here is our proposed model:

1. We captured real-time packets from internet using CICFlowMeter. Sniffed packets are as .pcap files and it was converted to .csv files in order to process it.
2. Data pre-processing is performed on the data.
3. Then it is passed to our Intrusion Detection Classifier. We have integrated three models into our classifier.
   - We have used Neural Network, LSTM and Random Forest to build three models.
   - Each of this model outputs their predicted probabilities.
   - We choose maximum probability from each model.
   - A voting system is used to choose one of the predictions and return it as output.
4. Based on classifier's output we predict whether the packet is an benign or some type of intrusion.

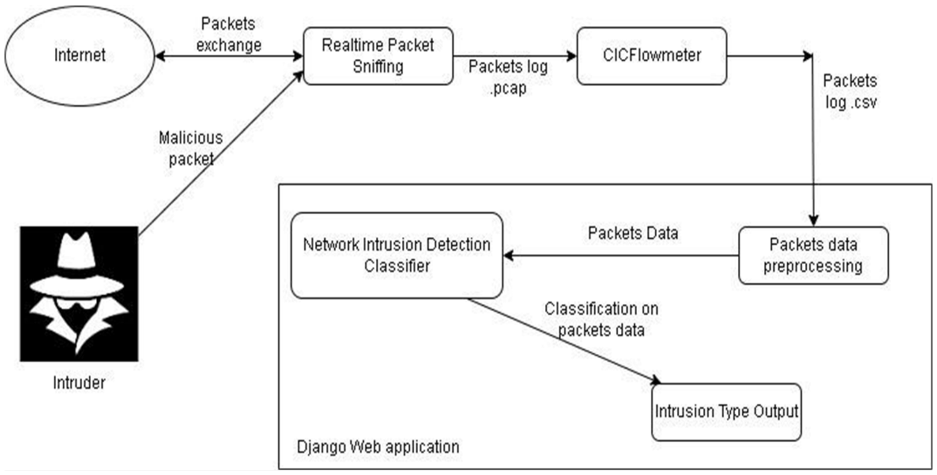


Figure 1: Our proposed model

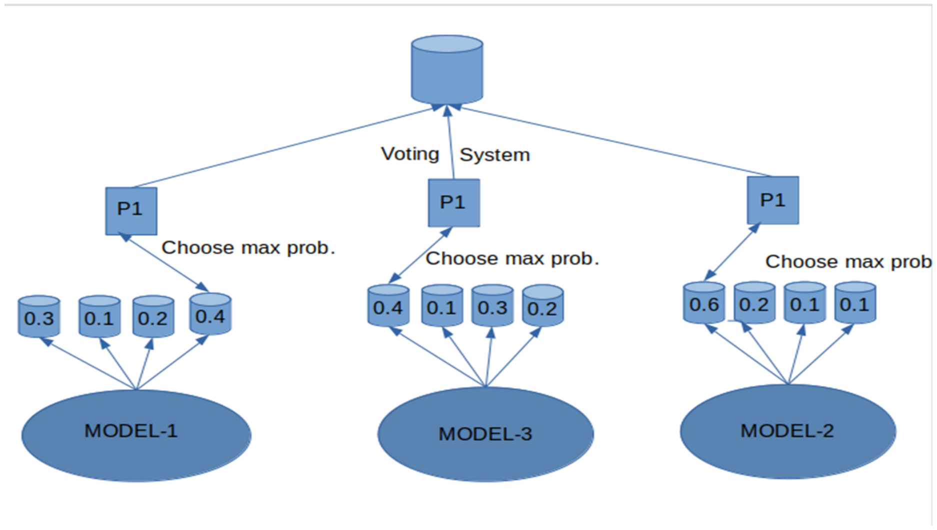


Figure 2: Our proposed model Architecture

3 Dataset Used

CICIDS2017[1] dataset contains benign and themostup-to-date common attacks, which resembles the true real-world data (PCAPs). It also includes the results of the network traffic analysis with labeled flows based on the time stamp, source, and destination IPs, source and destination ports, protocols and attack (CSV files). It contains all common available protocols, such as HTTP, HTTPS, FTP, SSH and email protocols. This dataset includes attacks such as:

1. DDoS
2. PortScan
3. Bot
4. Infiltration
5. Web Attack Brute Force, Web Attack XSS, Web Attack Sql Injection
6. FTP Patator, SSH Patator
7. DoS slowloris, DoS Slowhttptest, DoS Hulk, DoS GoldenEye
8. Heartbleed

4 Pre-Processing

Data Cleaning : Our dataset set contained some NULL/NaN values. As number of Null values is very small(0.01 % of dataset), we removed it. Then checked for non-finite values in our dataset and removed it. Many labels contained non-ASCII values which we changed to desired values. Also column names has extra leading and trailing spaces which has to be removed.

Feature Selection : Feature selection is the process of reducing the number of input variables when developing a predictive model. It is desirable to reduce the number of input variables to both reduce the computational cost of modeling and, in some cases, to improve the performance of the model. So we removed undesirable features from the dataset.

Data Normalization : The features dataset contains values that have different scale range during learning and minimize the loss function. These scales affect each iteration step during the gradient optimization process, which tends to affect the learning rate optimization, since we want the model to converge to the global or local minimum quickly and accurately. Min-Max normalization has several advantages over the standard scaling methods. Min-Max scaling has the ability to deal with the distribution of features that are not Gaussian, since the anomaly detection applications have no specific distribution to follow. The Min-Max normalization technique is proposed to prevent the gradient while optimizing the loss function from the un-smoothing path toward the global minimum. Here is the Normalization equation:

$$X\prime = \frac{X - X_{min}}{X_{max} - X_{min}}$$

SMOTE Analysis (Resampling) : Imbalanced classification involves developing predictive models on classification datasets that have a severe class imbalance. The challenge of working with imbalanced datasets is that most machine learning techniques will ignore, and in turn have poor performance on, the minority class, although typically it is performance on the minority class that is most important. One approach to addressing imbalanced datasets is to oversample the minority class. The simplest approach involves duplicating examples in the minority class, although these examples don't add any new information to the model. Instead, new examples can be synthesized from the existing examples. This is a type of data augmentation for the minority class and is referred to as the Synthetic Minority Oversampling Technique, or SMOTE for short.

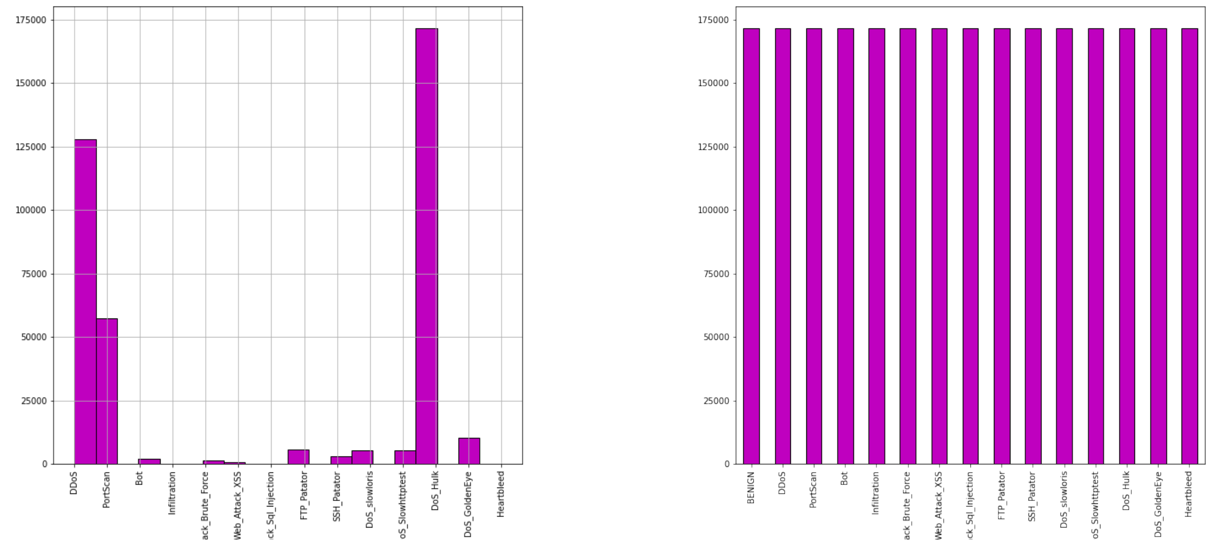


Figure 3: Intrusion types in our dataset before (left) and after (right) SMOTE analysis

5 Algorithms Used

5.1 Neural Network

In general, neural networks[2] are considered one of the most important computational networks, which consists of multiple hidden layers with nodes and ways of interconnecting nodes. The neural networks algorithm that builds one of the model of this project can be described via three main steps. First: the topology of the model, which describes the number of layers and neurons for each layer with the connections between them. Second: the forward propagation with its perceptron classifier and activation function used by the artificial neurons. Third: the back propagation with loss function and optimizer.

ReLU activation function : The rectified linear activation function(ReLU) is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero. It has become the default activation function for many types of neural networks because a model that uses it is easier to train and often achieves better performance.
Here is ReLU activation function:

$$f(x) = max(0, z) = \begin{cases} 0 & \text{if } z \leq 0 \\ z & \text{otherwise} \end{cases}$$

Softmax activation function : Softmax is a mathematical function that converts a vector of numbers into a vector of probabilities, where the probabilities of each value are proportional to the relative scale of each value in the vector. Each value in the output of the softmax function is interpreted as the probability of membership for each class.
Here is Softmax activation function:

$$\sigma_1(z_1) = \frac{e^{z_1}}{\sum_{i=1}^{k} e^{z_i}}$$

where k is no. of classes, $z_1 = \sum_{j=1}^{d} x_{ij} \; * \; w_{j1}$ silimarly for $z_2, z_3, ..., z_k$

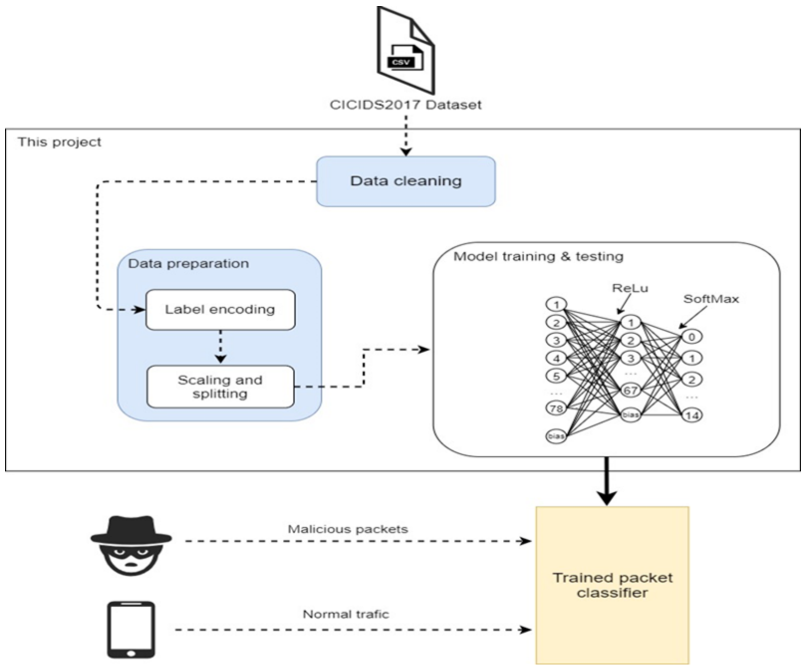


Figure 4: Our Neural Network Model

5.2 Hidden Markov Model

Hidden Markov Model[3] improves the accuracy of sequences of prediction when combined with regular base Classifier. Hidden states are drawn from the categorical distribution of classification class labels. State Transition Matrix are estimated from unlabeled data using the Viterbi Algorithm. Possible distributions are drawn from the categorical distribution of the classification class labels. Emission probabilities are estimated by the prediction probability estimates.

5.3 Long Short-Term Memory (LSTM)

Recurrent Neural Network (RNN) is a modified version of neural network that updates its internal state over time. By forming circular connections within the network, RNNs can memorize past inputs and capture temporal properties in sequential data. However, RNNs can retain memory for only a short amount of time steps due to the vanishing gradient problem. LSTM[4] solves the vanishing gradient problem by introducing three gates (input, forget and output) around special memory units called cell states c_t. The gates control the update of the cell states.

5.4 Random Forest

Random Forest[5] is a supervised ensemble machine learning algorithm. The ensemble approach is used when multiple machine learning algorithms are used and combined. Random Forest is constructed from multiple decision trees, which may achieve more accurate and stable results. Random Forest is trained using bagging methods; it can be used in classification and regression tasks, but here we are using it for classification task. It builds decision trees on different samples and takes their majority vote for classification. It has low classification error compared to other traditional classification algorithms. Number of trees, minimum node size and number of features used for splitting each node. Here are few advantages of using Random Forest:

- Random forests are extremely flexible and have very high accuracy.
- Random forests have less variance than a single decision tree.
- It also do not require preparation of the input data. You do not have to scale the data.
- It also maintains accuracy even when a large proportion of the data are missing.

6 Implementation

6.1 Training

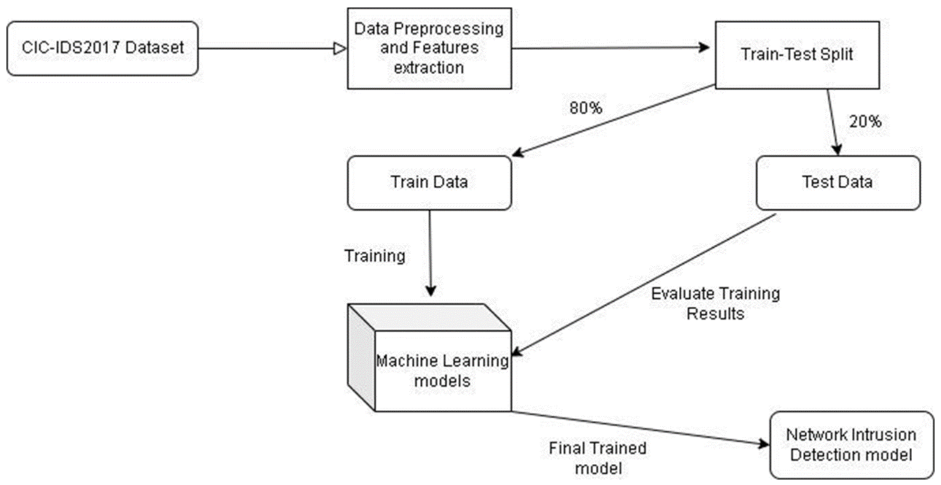


Figure 5: Our models training

Once data pre-processing is done we went ahead with training our different models. We prepared our training data and testing data in the ratio 80:20. Then we trained all of our three models. Parameters used for different models are mentioned below:

Neural Network Model Parameters:

- Train-Test split ratio: 80:20
- No. of nodes in Input Layer: 78 nodes
- No. of Hidden Layers: 1 hiddden layer with 67 nodes
- No. of nodes in Output Layer: 15 nodes
- Optimizer used: Adam
- Dropout rate: 0.2
- Activation function used: ReLu, Softmax
- Loss: Sparse Categorical Crossentropy
- No. of Epochs: 10

LSTM Model Parameters:

- Train-Test split ratio: 80:20
- No. of nodes in Input Layer: (39, 16)
- BatchNormalization: (39, 16)
- No. of Hidden Layers: 1
- BatchNormalization: (16)
- No. of nodes in Output Layer: 15
- Optimizer used: Adam
- Dropout rate: 0.2
- Activation function used: ReLu, Softmax
- Loss: Sparse Categorical Crossentropy
- No. of Epochs: 5

Random Forest Model Parameters:

- Max Depth = 2

6.2 Real time Packet Sniffing

We have implemented real time packet sniffing using CICFlowMeter. Internally, sniffed packets are stored in pcap format . Then these pcap files are read and stored into csv format by CICFlowMeter. Now this csv files is passed to our trained model to predict the attack type.

6.3 Real time Attack

We have simulated real time attack and tested our model if it was able to detect the attacks. We have simulated the attack in Ubuntu using hping3. The attack we performed is Dos(Denial Of Service) Attack. Denial of service attack (DoS), is an attack launched by a single attacker using his own computer and network by flooding the victim's server in order to shut down the target service. In most common cases, the attacker is simply sending partial requests in order to keep the connection open, over and over again until the server can not handle it any longer. If the server is overloaded with requests or connections, it is exhausted and can no longer accept any new connections.

7 Results

Once our models are built we have tested our models on testing data. Here are epoch Vs accuracy and epoch Vs loss graph for Neural Network and LSTM model.

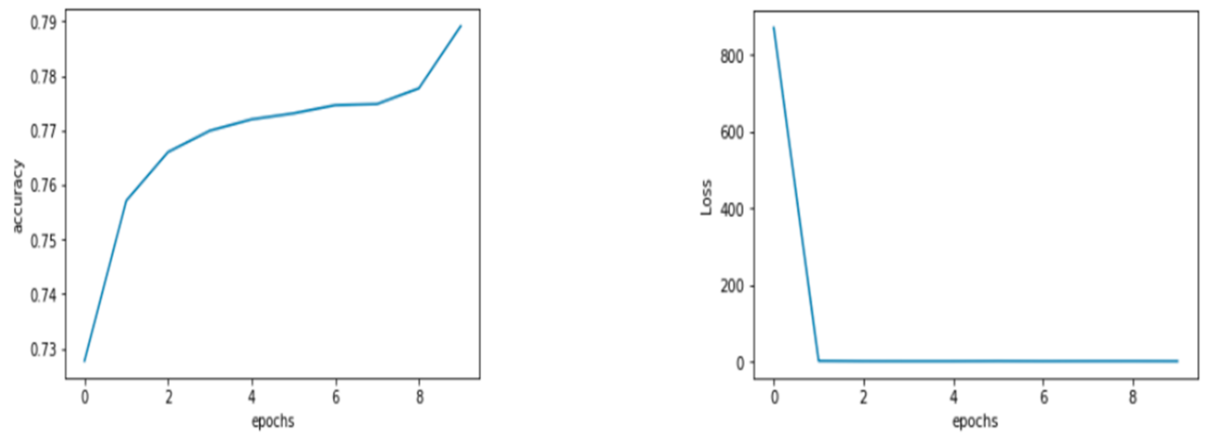


Figure 6: Epoch Vs Accuracy and Epoch Vs Loss graph for Neural Network Model

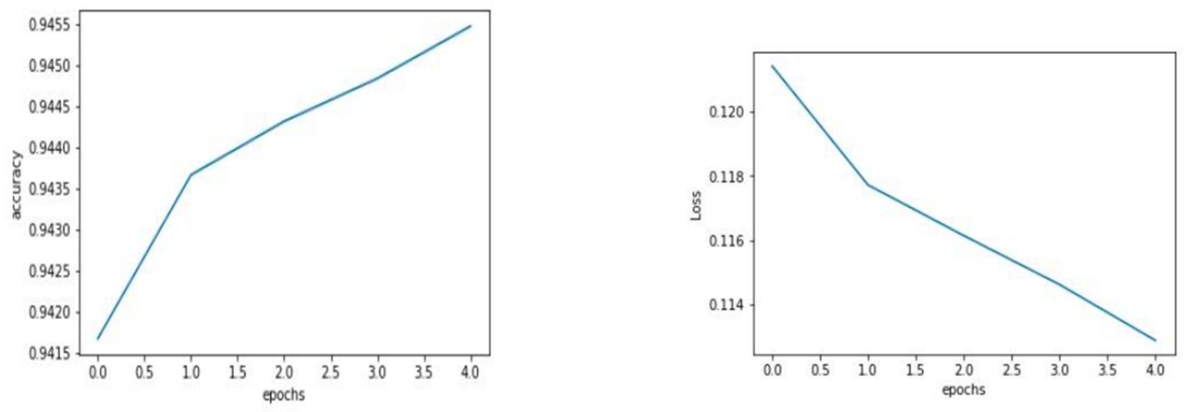


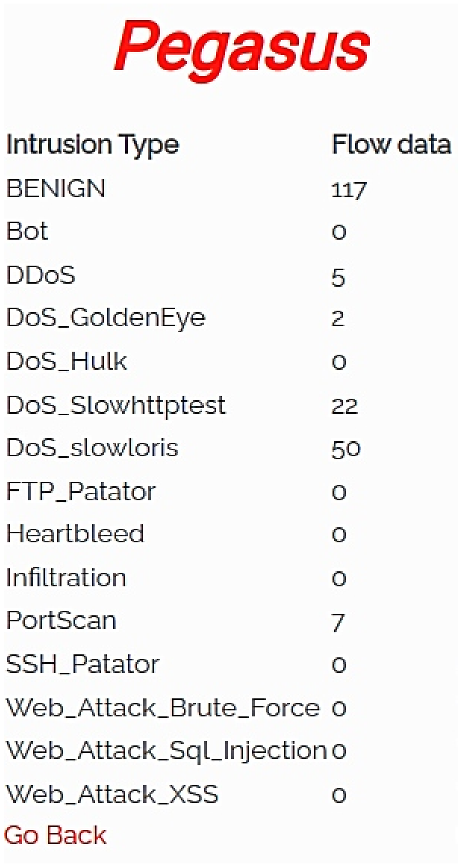
Figure 7: Epoch Vs Accuracy and Epoch Vs Loss graph for LSTM Model

Models	**F1 Score**	**Accuracy (%)**
Neural Network Model	0.71	81
LSTM Model	0.78	87
Random Forest Model	0.64	63
Ensemble Model	0.97	98

Table 1: Our Experimented model results

We have used F1 score and Accuracy as our evaluation metric for all the models. As shown in above table we have different model performances on our dataset. Out of all the three models we used, LSTM performed the best. But when we integrated all of these models together our model's accuracy reached 98 %.

Once our ensemble model was ready we went ahead and performed real time DoS attack. Our IDS model successfully performed and detected the intrusion types. As seen in the Figure 7 below it was able to detect different DoS attacks. There are few false positives but overall our ensembled IDS model performed well.



Pegasus

Intrusion Type	Flow data
BENIGN	117
Bot	0
DDoS	5
DoS_GoldenEye	2
DoS_Hulk	0
DoS_Slowhttptest	22
DoS_slowloris	50
FTP_Patator	0
Heartbleed	0
Infiltration	0
PortScan	7
SSH_Patator	0
Web_Attack_Brute_Force	0
Web_Attack_Sql_Injection	0
Web_Attack_XSS	0

Figure 8: Result of our IDS when a real time attack was performed

8 Conclusion

Our proposed model was to integrate Neural Network with Hidden Markov Model, LSTM and Random Forest models together into a IDS which can successfully detect different types of intrusion. Our ensembled model performed really well with an accuracy of 98 %. To test our models accuracy we performed real time attack on it using hping3 form Ubuntu and it was able to classify the intrusion correctly. We have achieved our goal of building a integrated IDS that performs well and gives less false positive alarms.

Limitations : Our model can correctly classify the packets intrusion type most of the time. But none of the Intrusion detection system can be totally accurate. There were few limitaions we found in our model:

- Some false positive detection is still there while predicting on Realtime sniffed packets.
- Not accurate enough to detect all Zero-day attacks.
- We have reached to our goal with certain flaws which can be rectified in our future work.

Future Work : In order to improve our model we perform some tasks in future. We can improve HMM Integration into Neural Network model to increase overall accuracy of ensemble model. We need to keep improving our model to have more categorization of malwares after analysing new attacks. We can also try to operate on realtime packet capturing for an organization which will help our model to be trained well.

References

[1] "Intrusion Detection Evaluation Dataset (CIC-IDS2017)." https://www.unb.ca/cic/datasets/ids-2017.html.

[2] "Neural Network)." https://www.techtarget.com/searchenterpriseai/definition/neural-network.

[3] R. Jain and N. Abouzakhar, "Hidden markov model based anomaly intrusion detection," pp. 528–533, 01 2012.

[4] "LSTM)." https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-expla

[5] "Random Forest)." https://towardsdatascience.com/understanding-random-forest-58381e0602d2.