Implementation of threats detection modeling with Deep learning in IoT botnet attack environment

Kanti Singh Sangher[1+], Dr. Archana Singh[2], Dr. Hari Mohan Pandey[3] and Dr. Lakshmi Kalyani[4]

[1] School of IT, Centre for Development of Advanced Computing, Noida-201307, INDIA, kantisingh@cdac.in
[2] Amity School of Engineering & Technology,Amity University, Noida- 201313, INDIA, asingh27@amity.edu
[3]Department of Computing and Informatics, Bournemouth University, Fern Barrow, Poole BH12 5BB United Kingdom, profharimohanpandey@gmail.com
[4] School of IT,Centre for Development of Advanced Computing, Noida-201307, INDIA, lakshmikalyani@cdac.in

Abstract: IoT forensics where security and privacy are the key concern as the data the majorly hold personal information. So how to work on the vulnerabilities available from the IoT environment and classify them to get the best results to perform the forensics is covered in the paper. In IoT forensics botnet dataset analysed using deep learning classification to get the understanding that how deep learning can be used effectively for forensic analysis. So, research work provides advanced digital forensics methods i.e., collection of evidence and analysis of dataset for IoT forensics implementation. Since a decade ago, we are seeing a reality where hacking into a client's PC utilizing small bots or blocking a gathering of interconnected gadgets is not any more unthinkable. These little bots are called botnets (e.g., Mirai, Torii and so on.), which are a gathering of deadly codes that can obstruct the whole security. As Internet of Things (IoT) is developing quickly, the interconnected gadgets are helpless to penetrate as one influenced gadget can crumple the entire system. As Internet of Things (IoT) is developing quickly, the interconnected gadgets are defenseless to break as one influenced gadget can hamper the entire system. The security danger stays as botnet assaults increment their essence to the interconnected gadgets. In this work, we are proposing a novel correlation between AI (SVM and KNN) and profound learning draws near (Neural system) to discover which approach creates better outcome while learning the assault designs. Research explores the IoT forensics analysis. In IoT forensics models were applied on a composite information storehouse which was made by consolidating the outcomes found from the examination we did on Torii botnet test, with the CTU-13 dataset of botnet assaults on IoT environment.

Keywords: Network forensics, deep learning, cyber threat intelligence, botnets, DeBot, interference problem, Internet of Things (IoT), Torii.

1. Introduction

The focal point of this research work is finding the artifacts in upcoming advanced areas of digital forensics and analyzes how to resolve the crimes. The research work [4] focuses on network traffic. It works on HTTP response manipulation and relates client –server traffic data set to the user. Many investigations of Mirai to date have focused on a traditional malware analysis of the executable code found on infected IoT devices, which can be collected from an infected device or a honeypot [1,2,3].

However, from a forensic investigator's perspective, the executable for infecting an IoT device is not the only area worth exploring in a Mirai network [4]. IoT forensics and research to detect the botnet attack will be very helpful to prevent the innovations which are making the world technically strong and vulnerable too.

2. Materials, Method & Experimental Approach

2.1 Digital Forensics Analysis: Finding the vulnerabilities of overall system

The attacker uses botnets to implement a virtual framework to collapse the IoT network. For example, a network of CCTV cameras which are interconnected through a series of IoT connections are vulnerable to this type of attack, which may result to crucial information manipulation. The figure 1 above shows the statistics of botnet attacks in 2018 around the world. In this work, we are doing a deep analysis of the recent Botnet "Torii" using Deep learning to find out its characteristics and behavior.

In this work, we are using the comparison of machine learning approaches with the deep learning approach to find out the how deep learning does a more accurate job than machine learning. The methodology of this work is divided into following steps as shown in the Fig 1:

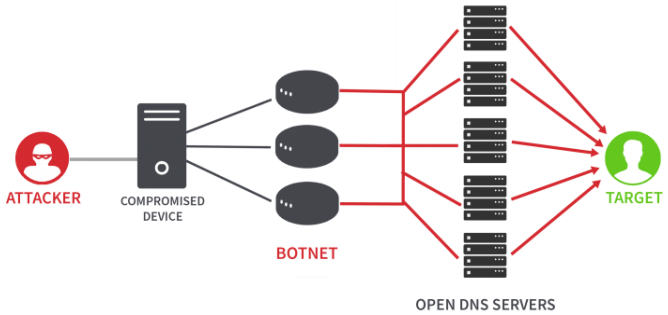

Fig 1: Attack using Botnet in IoT environment
(Source:https://www.4d-dc.com/insight/future-of-ddos-attacks-dns-amplification-2019)

We need to enter a loopback address (for e.g., 127.0.0.1) to start the server. The loopback address will then be configured into the local server and the fake network will be created to prevent the botnet from recognizing it is running in VM (Virtual Machine). Wireshark is a network traffic analysis tool which was previously known as Ethereal. It analyzes the packets between the botnet and the C&C server and shows them in human readable output. includes several features like filters and color coding etc. that makes it easy to analyze the network communication. We are using Wireshark to capture the communication traffic logs when the botnet is trying to communicate with the Botmaster through the C&C server. The steps to capture the network logs are as follows:

1. Select Capture Interfaces.
2. Select the interface on which packets need to be captured. This will usually be the interface where the Packet/s column is constantly changing, which would indicate the presence of live traffic).
3. Click "Start" to initiate the capture mode.
4. Restart the situation. The capture mode should show the number of packets increasing.
5. Avoid running any other internet applications while capturing, closing other browsers, Instant messengers etc.
6. Once the necessary environment is setup and the processes are regenerated, stop the services. It may take a while to show the results.
7. Save the traced output in a .pcap format which can be analyzed further. It saves the output in libpcap format.

2.2 Feature extraction

When the above processes are done, the behavioural snapshots are taken of the host and the protocol that are used by the packets to communicate with the attacker. The snapshot contains all the traffic statistics [5] over several characteristics and behavioural parameters to collect all the incoming and outgoing traffic. The same IP

generated packets are being captured and the protocols are analysed by extracting from the data gathered. The source and destination IPs and the TCP/UDP protocol transmissions are also analysed.

2.3 Training and testing phase

We are using Anaconda v3 to setup the training environment of the CTU-13 dataset. Using Jupyter Notebook, we are using ipython to implement the environment. After the training of the dataset is dine, it splits the dataset and categorizes into training_set.dat and testing_set.dat. The training set includes the classification information and the DeBot model is applied on this set. The testing set validates the training model using cross validation. Once the validation is done, the model becomes ready for ne data input. We used joblib library package of python to call the stored model to new data.

The dataset that we have used for this purpose is the CTU-13 botnet dataset[6]. This dataset was captured with the intent to capture a large traffic in the network used by the botnets. This dataset was created at the CTU university. It consists of 13 phases of execution. On each phase, a sample of malware was analyzed, and the parameters were extracted, and the feature set were stored like protocols used etc. The phases were captured in pcap file formats. The pcap files contain the netflow, weblog information extracted from network capture using tools like Wireshark. The netflows are divided into unidirectional and bidirectional netflows. The advantages of the bidirectional netflows are as follows: 1. They have more detailed class labels. 2. They store more information, 3. They solve the differentiation between the client and server side.

The initiating process of ApateDNS to start a fake network.Capturing of inbound and outbound packets used by the botnet to communicate with C&C server[7]. The results of Process monitor (Procmon) which indicates about the processes spawned by the botnet. From the analysis it was found that the Torii botnet is using an anti-VM trick to check whether it is running in a virtual environment or not. If the botnet recognizes that it is inside a virtual environment, it triggers the anti-VM trick to prevent the VM from recognizing and analyzing the botnet. To mitigate this situation, it is necessary to create a fake network. The tools that are being used for this analysis are as follows:

ApateDNS: It creates a fake network using a loopback IP address. **Wireshark:** It captures the incoming and outgoing network packets from and to the C&C sever. **Procmon:** Procmon or Process Monitor is a freeware by Microsoft which shows the processes created by the botnet. ApateDNS application used to create a fake network. User can provide IP address of their choice to loopback the network connection. Here, we have provided the loopback address 127.0.0.1 to create an intra-loop of local DNS. By clicking start server, it initiates the fake network server.

The feature set found from the analysis of Torii botnet are as follows:

- Registry path where it is infecting the system (mostly, HKEY_LOCAL_MACHINE, HKEY_CURRENT_USER and HKEY_USER).

Strings used for spawning processes. Processes used (both legitimate and spawned) to gain access to the privileged administration framework. Attack patterns and parameters used while spreading in the network.

3. RESULTS

All these features are extracted and then combined with the CTU-13 dataset of botnet attacks to create a composite data repository. Once the composite dataset is ready, we then applied two phases of experiments on the same data repository. In the first phase, we applied two machine learning approaches – Support vector Machine (SVM) and K-Nearest Neighbor (KNN)algorithms to find out the accuracy level. We also noticed the processing time for both the algorithms. In second phase, we applied deep learning approach -neural network sequential model and found out the accuracy and processing time by applying it on the same dataset i.e. CTU-13. We applied the NN(S) algorithm on the dataset to train the algorithm about the patterns and parameters. The model is then created

(sequential model) and the parameters are fed into the system to enable to it detect new unseen data. It imported Keras and sequential & dense modules from Keras to support the deep learning training algorithm in this thesis. In the below figure, we are importing the necessary packages and libraries by calling them into Jupyter Notebook's local repository. The following screenshots show the results found by applying machine learning approaches (SVM and KNN) on the CTU-13 dataset.

The results found shown in Fig 2, indicates that SVM has an accuracy of 0.6140301582841622 and KNN has accuracy of 0. 9326589866067247.

```
In [75]: #implementing Support Vector Machine (SVM) algorithm and model

model = svm.SVC(gamma=1)
model.fit(train_X, train_y)
prediction = model.predict(test_X) #prediction based on testing_set
print('The accuracy of the SVM is: ', metrics.accuracy_score(prediction, test_y))
expected = train_y #expected result stored in training_set
predicted = model.predict(train_X)
print('Classification report:')
print(metrics.classification_report(expected, predicted)) #calculating classification report
print('Confusion matrix:')
print(metrics.confusion_matrix(expected, predicted)) #calculating confusion matrix

The accuracy of the SVM is:  0.6140301582841622
Classification report:
             precision    recall  f1-score   support

          0       1.00      1.00      1.00        67
         21       1.00      1.00      1.00       131
         25       1.00      1.00      1.00      4148
         53       1.00      1.00      1.00      2812
         80       1.00      1.00      1.00      1428
         81       1.00      1.00      1.00         2
         82       1.00      1.00      1.00         2
         88       1.00      1.00      1.00         3
        137       1.00      1.00      1.00        38
        138       1.00      1.00      1.00         8
        139       1.00      1.00      1.00        16
        161       1.00      1.00      1.00        10
        443       1.00      1.00      1.00     15144
        587       1.00      1.00      1.00        81
        888       1.00      1.00      1.00        14
       1057       1.00      1.00      1.00         1
       1085       1.00      1.00      1.00         1
```

```
# Adding a output layer.

model.add(Dense(1, init='uniform', activation='sigmoid'))

# Compiling the model

model.compile(loss='binary_crossentropy',

              optimizer='adam', metrics=['accuracy'])

# Fitting the model

model.fit(X, Y, nb_epoch=50, batch_size=10)


scores = model.evaluate(X, Y)

print("%s: %.2f%%" % (model.metrics_names[1], scores[1] * 100))
```

```
Using TensorFlow backend.
```

Fig 2: Neural network sequential model implementation

The sequential model has been trained with the composite CTU-13 dataset and the attack patterns were trained once through 50 iterations of epochs (epoch 1 to epoch 50). As the results indicated, if we use less than 50 epochs, the model underfits. If we use more than 50 epochs, the model overfits. An epoch contains a full forward and backward passing of the whole dataset through the neural network. Based on the number of epochs, the model becomes underfitted, optimal or overfitted. An underfitted model is the one which not able to predict the data accurately. An optimal model is the one which predicts the data correctly and an overfitted model[8] is the one which has been trained too well based on the parameters given, that it doesn't consider any new unseen data for prediction. Precision and Recall helps further to get the understanding of how strong the accuracy given holds true for a particular problem. For making our model an optimal one, we adjusted the number of epochs to 50 which

gives an accuracy of 99.99%. So the accuracy lies as per the table 1 always less than 1. Comparison between the various machine learning and deep learning analysis are shown in Fig 3.

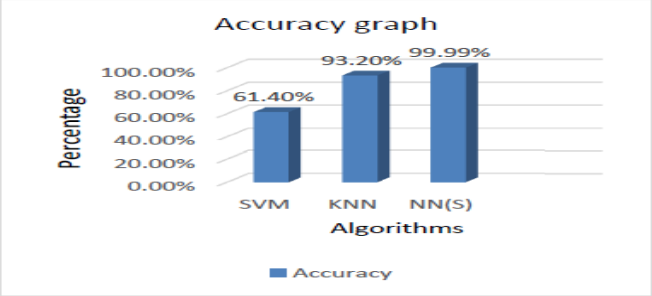

Fig 3: Comparison between machine learning and deep learning results

The table 1 shows the accuracy levels of the machine learning algorithms with deep learning algorithms.

Table 1: Accuracy graph

Model Name	Accuracy	Processing time	Computational complexity
*Support Vector Machine(SVM)	0.6140301582841622	72 seconds	O(max(n.d)) min(n,d)^2
*K-Nearest Neighbor(KNN)	0.9326589866067247	61 seconds	O(nd+km)
*Neural Network sequential algorithm	0.9999	55 seconds	O(nd)

In machine learning Over-fitting occurs when a model fits the training data too well and as a result can't accurately predict on unseen test data. So, the model has simply memorized specific patterns and noise in the training data, but is not flexible enough to make predictions on real data. But in our research precision ad recall was set to 1 and data results received were different accuracy as per the Table 1.

4. Conclusion

The world already saw how deep learning can improve the quality of society; it is one of the most promising technologies against botnet attacks. In this work, we proposed a novel comparison between machine learning and deep learning algorithm by implementing SVM, KNN and Neural network models on the same CTU-13 dataset. To show the capacity of our created classification set to identify new varieties of botnets, a changed adaptation of Torii code sequence will be utilized to produce a new dataset repository and will be looked at existing mark and stream via location strategies.

5. Future Scope

The future scope of the present work can be that the composite dataset created can be further enhanced with regularly released botnet parameters and recursive algorithms can be used for increasing the epoch count to pass the data gathered forward and backward more frequently and to increase level of accuracy. So, in future we will try to embed the dark web dataset and deep learning algorithms to find out the illegal activities within the dark

web which contributes to IoT attacks. As per the finding of dark web vulnerability and acquisition path, in future it will be extended to get the threats and detection of illegal activities using cyber threat intelligence [9]. Analysis based on the dataset for IoT further can be explored to enhance the IoT forensics with help of deep learning algorithm [10] for different devices used for smart application. So overall digital forensics in advanced areas can be improved and the acquisition and analysis of the data with vulnerabilities exploitation and deep learning will be great area to work.

Acknowledgements

We would like to acknowledge Pramtesh Majumdar for providing materials, tools and guidance.

6. References

[1] G. Kambourakis, C. Kolias and A. Stavrou, "The Mirai botnet and the IoT Zombie Armies," MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM), 2017, 267-272.

[2] J. Margolis, T. T. Oh, S. Jadhav, Y. H. Kim and J. N. Kim, "An In-Depth Analysis of the Mirai Botnet," *2017 International Conference on Software Security and Assurance (ICSSA)*, 2017, 6-12.

[3] Wang, W. Chang, S. Chen and A. Mohaisen, "Delving Into Internet DDoS Attacks by Botnets: Characterization and Analysis," in *IEEE/ACM Transactions on Networking*, vol. 26, no. 6, . Dec'2018, 2843-2855.

[4] C. D. McDermott, F. Majdani and A. V. Petrovski, "Botnet Detection in the Internet of Things using Deep Learning Approaches," *2018 International Joint Conference on Neural Networks (IJCNN)*, 2018, 1-8,

[5] Yan, Q., Yu, F. R., Gong, Q., & Li, J. (2018). Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges. IEEE Communications Surveys & Tutorials, 18(1), 602-622.

[6] P. Torres, C. Catania, S. Garcia and C. G. Garino, "An analysis of Recurrent Neural Networks for Botnet detection behavior," *2016 IEEE Biennial Congress of Argentina (ARGENCON)*, 2016, 1-6.

[7] Roosmalen, J. V., Vranken, H. P. E., van Eekelen, M. C. J. D., & Haddad, H. H. (2018). Applying Deep Learning on Packet Flows for Botnet Detection. In Haddad, HH (ed.), SAC18: The 17th edition of the Computer Security track at the 33rd ACM Symposium on Applied Computing, 9-13 April 2018, Pau, France,1629-1637.

[8] Kudugunta, S., & Ferrara, E. (2018). Deep neural networks for bot detection. Information Sciences, 467, 312-322.

[9] N. Koroniotis, N. Moustafa and E. Sitnikova, "Forensics and Deep Learning Mechanisms for Botnets in Internet of Things: A Survey of Challenges and Solutions," in IEEE Access 2019, vol. 7, 61764-61785.

[10] H. Saleh, S. Agaian and K. Mohamamd, "Digital forensics: Electronic evidence collection, examination and analysis by using combine moments in spatial and transform domain," 2009 IEEE International Conference on Systems, Man and Cybernetics, San Antonio, TX, USA, 2009, 3489-3494.