VOL. 12, NO. 12, JUNE 2017 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences


www.arpnjournals.com

INTRUSION DETECTION SYSTEM USING BIG DATA FRAMEWORK

Abinesh Kamal K. U. and Shiju Sathyadevan
Amrita Center for Cyber Security Systems and Networks, Amrita School of Engineering, Amritapuri, Amrita Vishwa Vidyapeetham
Amrita University, India
E-Mail: abineshjerry@gmail.com

ABSTRACT

In the enormous stream of network traffic, there is no way to identify which packet is benign and which is an anomaly packet. Hence, we intend to develop a new network intrusion detection model using apache-spark to improve the performance and to detect the intrusions while handling the colossal stream of network traffic in IDS. The model can detect known intrusion effectively using real-time analytics and hence identify unknown data schema compared with traditional IDS. The objective of the model addresses the following capabilities: Deep Packet Inspection (DPI) by inspecting the network traffic and examining the properties that describe the intrusion characteristics. Collaborating the vulnerability assessment with human intervention, using C.45 decision tree algorithm, optimizes pattern matching to boost detection rate. The clustered hosts are grouped based on their number of visits in an unique IP. The intrusion classifiers are developed by investigating each IP groups which reflects different properties used for prediction. The prediction model is built over Amrita Big Data Apache-Spark framework as a sequence of workflows. The above workflow is implemented in Amrita Big Data Framework (ABDF) to improve the detection time and performance, the model output provides effective results in detecting DOS attacks and port scanning attacks.

Keywords: apache spark, data mining, intrusion detection, network security.

INTRODUCTION

Currently, the key research in the field of data mining has numerous shortcomings as follows: Data produced automatically integrates only small amount of intrusion characteristics, therefore rulemaking becomes inefficient. Most of the researchers go after Wenke Lee and Portnoy, because of their work upon integration of data mining algorithms such as fuzzy, genetic engineering.[1-4].Even Though analysis in the automatic rule extraction is elementary, the data mining detection technique are not viable to identify new threats automatically. Another major factor is the evaluation criteria of the technique that includes the performance measurement and comparison of the normal behaviour of the system. Currently, accuracy and responses of any intrusion detection system (IDS) is not coherent and does not meet the demand of practical application.

Hence with the prior knowledge in Information security and data mining techniques, specific attributes are been carved and matched up with the network communications that reflect to the rules effectively. By registering apt data mining algorithm merging with appropriate anomaly detection together we deploy an efficient data-mining model based on network intrusion detection in apache-spark to improve the traditional methods.

METHOD DESCRIPTION

A flawless IDS integrated with different detection techniques can hit up different types of intrusion. The major approach of this research paper is to distinguish legitimate and non-legitimate intrusions and their performance evaluation within the system on a criteria of Port Scanning and Deny of Service (DoS) [5-7]. Based on above methods we designed and developed a intrusion detection model deployed in apache spark to improve the performance. The model is built by using training data which reflects the intrusion properties. To build the prediction model DARPA 1998 data set is used. This model detects new intrusions based on the classified training data. This paper dispute that the traffic source generated from various machines will have different properties with different patterns. On a server system, the traffic flow generated will be autonomous and does not send connection request in invoking. The workstation, host machine will act as an initiator of the TCP connections, the aim of the host machine is to obtain data. If a workstation machine receives huge volume of data suddenly then its behaviour is quietly abnormal and it should be monitored. To a server machine the huge number of concurrent active connection is quite normal, but if it exceeds the upper limit of the connections it should be also noticed. The decision of normal and abnormal activity is will be always unique. Each host will reflect a unique behaviour pattern as the services and streaming network traffic of hosts are different, which leads to expensive data processing. By knowing the above understandings this model verifies each and every requests based on unit time of the host in the network, then they are grouped into clusters based on their IP and visits time by Group - By features in the model framework[8].Each IP set will reflect different behaviour pattern thus we can avoiding the individual differences' set generated helps to define how the data is distributed over the network, and the distributed training data will authorize their own classifiers by using C.45 decision tree algorithm.

FRAMEWORK MODEL

The model framework [Figure-1] includes sequence of workflows, each workflow n takes input from the n-1 workflow and pass the output to the n+1th workflow. ABDF includes several processing modules

such as Hadoop, Spark, GPGPU algorithms for large scale computing. It also have several process elements together with various file adapters such as KAFKA, flat file adapter, the data to the ABDF framework can be streaming data or an static file. Our experiments work on streaming data with the help of ABDF apache spark framework. [9]

Data capturing by KAFKA

Network streaming data are collected from the core switch by KAFKA Pcap adapter in the framework.The network data are collected by an window size of 10. After collecting the data they are converted into Resilient Distributed Datasets (RDD).This form the input to the process flows(PF).

Data preprocessing using ABDF preprocessor

Streaming network data include all the data that are transmitted over the network, our aim is to extract the necessary information that reflects the intrusion properties from the network packet. Once the data are converted into RDD we can select the necessary fields that are given input to the next PF.

Table-1. Feature selection.

Fields selected for net PF
Date, Duration, Req_time, Service_Name, Source_Port, Destination_Port,Source_IP, Destination_IP.

Parameters calculation

Eight different parameters are calculated to transform the selected fields in a meaningful way[10] in the References section include:

- soip_difft_deip : get the total count value of number of source IP connected to a different destination IP by using SQL Runner Query [Table-2] feature in the framework.
- soip_difft_deport number: get the total count value of number of source IP connected to a different destination hosts port by using SQL Runner Query [TABLE III] feature in the framework.
- deport_same_soip number: get the total count value of number of connections established by destination port to the same source IP based on the destination host window time "w"
- deport_difft_soip number: get the total count value of number of connections established by destination port to the different source IP based on the destination host window time "w"
- The ratio of soip_difft_deip number based on source IP to all the total connection of different destination IP in a time window size "w".
- The ratio of soip_difft_deport number based on source ip to all the total connection of different destination port in a time window size "w".
- The ratio of deport_same_soip number based on destination port to all the total connection of same destination in a time window size "w".
- The ratio of deport_difft_soip number based on destination port to different source ip to all the total connections in a time window size "w".

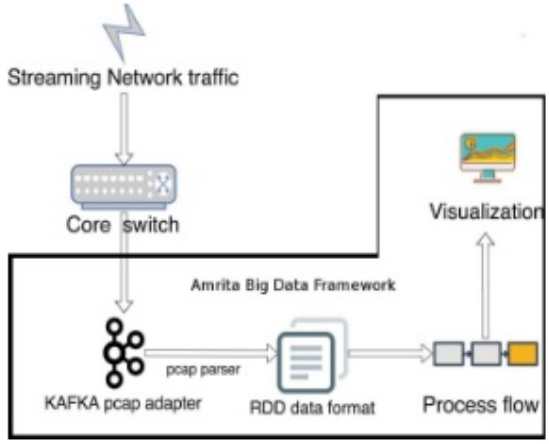

Figure- 1. Intrusion detection model workflow.

Table-2. SQL runner query for SOIP_DIFFT_DEIP.

Fields selected for net PF
select Date,Duration,Source_Port,Destination_Port,count.Source_IP,Destination_IP,Attack_check,Attack_Description,count.sip_diff_dip,count.sip_diff_dport from SqlRunner1 join (select Source_IP,count(Distinct Destination_IP) as sip_diff_dip,count(Distinct Destination_Port) as sip_diff_dport from SqlRunner1 Group By Source_IP) as count ON count.Source_IP=SqlRunner1.Source_IP"

Table-3. SQL RUNNER query for SOIP_DIFFT_DEPORT.

Fields selected for net PF
select Date,Duration,Source_Port,count.Destination_Port,Source_IP,Destination_IP,Attack_check,Attack_Description,sip_difft_dip,sip_difft_dport,count.dport_difft_sip from SqlRunner2 join(select Destination_Port,count(Distinct Source_IP) as dport_difft_sip from SqlRunner2 group by Destination_Port) as count ON count.Destination_Port=SqlRunner2.Destination_Port

Rule generation

To build the rule used DARPA 1998 dataset which have the details of intrusion properties with attack

description. The DARPA 1998 dataset includes details as shown in Table-4.

Table-4. Training data for model building.

DARPA 1998 dataset field selection
Date, Duration, Req_time, Service_Name, Source_Port, Destination_Port,Source_IP, Destination_IP, Attack_check, Attack_Description.

To generate the model, we need to build some rules that define how the model should identify the intrusions in the network traffic. Rules are built by using C 4.5 decision tree algorithm [11]. From the training data the eight parameters are calculated along with the attack description. The packet with intrusion are defined with attack description "normal" and the packet with normal payload are defined with "normal", whereas [Table-5],[Table-6] shows the rule generated by using C45 decision tree algorithm and the rule is generated in Java Simple Object Notation(JSON) format.

Table-5. Rule generation for DoS.

Rule for Port Scanning
{"tree":{"attributeIndex":7,"nodeName":"ratio_dport_same_sip","isLeaf":false,"children":[{"attributeIndex":7,"attributeValue":"\u003c\u003d0.035714285714285 71","children":{"attributeIndex":5,"nodeName":"ratio_sip_diff_dip","isLeaf":false,"children":[{"attributeIndex":5,"attributeValue":"\u003c\u003d0.32142857","children":{"attributeIndex":0,"nodeName":"normal","isLeaf":true},"mean":0.32142857},{"attributeIndex":5,"attributeValue":"\u003e0.32142857","children":{"attributeIndex":0,"nodeName":"port-scan","isLeaf":true},"mean":0.32142857}]},"mean":0.03571428571428571},{"attributeIndex":7,"attributeValue":"\u003e0.03571428571428571","children":{"attributeIndex":0,"nodeName":"port-scan","isLeaf":true},"mean":0.03571428571428571}]},"modelName":"Portscanc45","metrics":{"accuracy":100.0,"recall":50.0,"precision":50.0,"fmeasure":50.0}}

Table-6. Rule generation for port scanning attack.

Rule for DOS attack
{"tree":{"attributeIndex":6,"nodeName":"ratio_sip_diff_dip","isLeaf":false,"children":[{"attributeIndex":6,"attributeValue":"\u003c\u003d0.0043993494","children":{"attributeIndex":0,"nodeName":"normal","isLeaf":true},"mean":0.0043993494},{"attributeIndex":6,"attributeValue":"\u003e0.0043993494","children":{"attributeIndex":0,"nodeName":"neptune","isLeaf":true},"mean":0.0043993494}]},"algorithmName":"decision_tree","modelName":"PF_ACK_model_C45","metrics":{"accuracy":100.0,"recall":50.0,"precision":50.0,"fmeasure":50.0}}

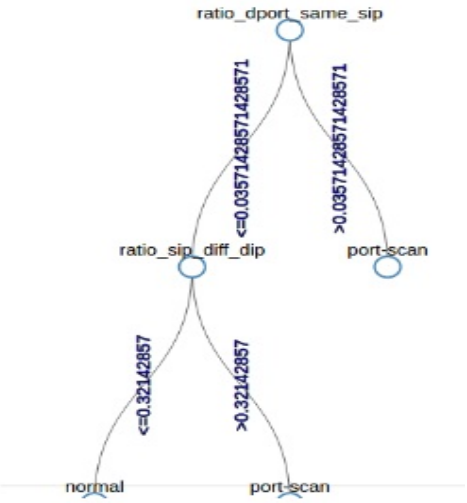

Figure-2. Tree generated for port scan based on the rule build by C45 algorithm.

EXPERIMENTAL SETUP

To perform the real-time attack tools such as hping3, wireshark, netcat are used. Tools like hping3 is used to perform DOS attack, netcat is used for performing port scanning and wireshark is used to collect the packets in the network.

Port scanning attack is done by using netcat command the syntax of netcat port scanning is "nc -v -z host port-range". Options in nc commands -z : Port scanning mode i.e. zero I/O mode. -n : Use numeric-only IP addresses i.e. do not use DNS to resolve ip addresses. -v : Be verbose [use twice -vv to be more verbose]. -w 1 : Set timeout value to 1.Command used to execute the attack to the target machine is Nc -v -z 10.30.56.3 10 -10000, this command will scan ports from 10 to 10000.During this operation the stream of packets are passed through the ABDF apache spark framework and the result [Figure-2] is viewed in Amrita Dashboard.

DoS attack is done by using the hping3 tool, Command used to perform SYN Flood (Neptune) attack. "Sudo hping3 -i u1 -S -p 80 -c 1000 10.30.56.3 ".The above query will send TCP SYN packets to 10.30.56.3, in Linux sudo query is needed because the hping3 tool generates raw SYN packets to attack the host.S - defines SYN flag, P 80 - defines the host port is 80, i u1 - defines wait for 1 ms time interval between transmitting packets, C - defines the total number of packets to be send and receive.[12][13]

PERFORMANCE EVALUATION AND RESULTS

The aim of this paper is to detect Intrusion such as DoS attack and Port scanning attack using Apache spark framework to increase the detection performance in large stream of network traffic. The process flow is built over apache framework. The response time to detect port scanning and DoS attacks are recorded in the framework and the analysis is given in Figure-2.

Table-7. Sample results obtained DoS attack- after parameter calculation.

soip_diff t_deip	soip_diff t_deport	deport _same_ soip	deport_di fft_soip	Attack Flag
4.5683	0.004678	9.1367	4.5683	DoS
4.5683	0.004678	9.1367	4.5683	DoS

Table-8. Sample results obtained port scanning - after parameter calculation.

soip_diff t_deip	soip_dif ft_depo rt	deport_ same_so ip	deport_di fft_soip	Attack Flag
0.002976	0.48214	9.1367	0.00297	port Scan
0.002976	0.47910	0.00297	0.00297	Port Scan

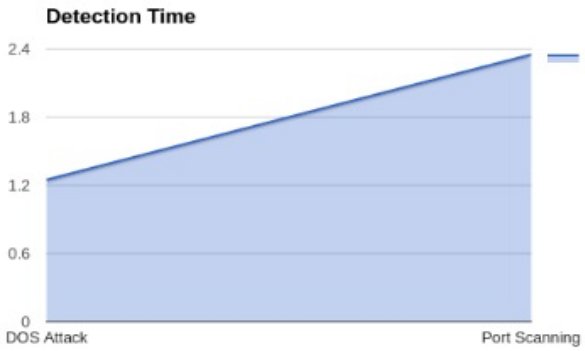

Figure-3. Performance analysis for prediction in dashboard.

From Figure-2 we can understand the total time taken to detect DoS attack and Port Scanning in Apache spark Framework.

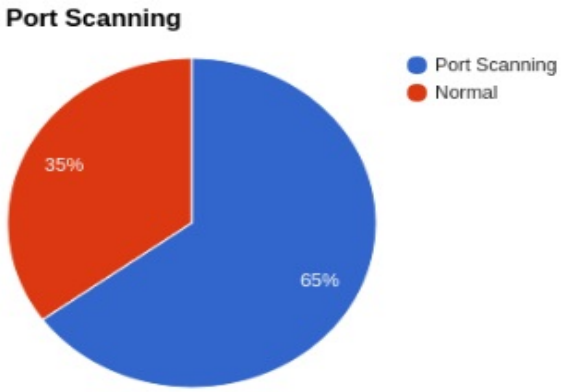

Figure-4. Output of port scanning prediction in dashboard.

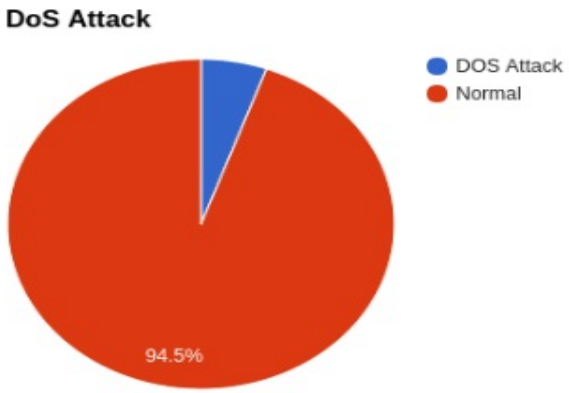

Figure-5. Output of DOS attack prediction in dashboard.

CONCLUSIONS

Experimental results show that in data mining IDS system, previous Knowledge about information security is needed. Data mining techniques can able to extract rules from the raw data that are in specific format, but the rules derived by the rules need not to be reasonable. Moreover, with the help of our framework in the experimental setup provides High performance, detection rate and the execution time is low as the framework is built over apache spark.

ACKNOWLEDGEMENTS

At the outset, I express my heartfelt gratitude to Shiju Sathyadevan and other faculties for their valuable guidance, timely suggestions and help in the completion of this paper. I express my heartfelt gratitude to all the staffs of the department of Amrita Cyber security systems networks, Amrita School of Engineering. I also thank the Evaluation panel for their valuable feedback, which made it possible in completing my paper. I thank all my friends who have helped me directly or indirectly, in the completion of this paper.

REFERENCES

[1] T. F. Lunt, R. Jagannathan, and R. Lee, "IDES: The Enhanced Prototype-A Real-Time Intrusion-Detection Expert System", Number SRI-CSL-88- 12. Computer Science Laboratory, SRI International, Menlo Park, CA, 1988.

[2] Wenke Lee and Salvatore J. Stolfo, "Data Mining Approaches for Intrusion Detection", In Proceedings of the 7th USENIX Security Symposium, 1998.

[3] Lee,Wenke, Salvatore J. Stolfo, and KuiW. Mok. "A data mining framework for building intrusion detection models." Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on. IEEE, 1999.

[4] W. Lee and S. J. Stolfo. A framework for constructing features and models for intrusion detection systems. ACM Transaction on Information and System Security, 3(4):227-261, Nov. 2000.

[5] P.Porras, D. Schnackenberg, *et al.* "The Common Intrusion Detection Framework Architecture, CIDF", University of California, 1998.

[6] Z. A. Othman, A. Bakar and I. Etubal, Improving signature detection classification model using features selection based on customized features. Intelligent Systems Design and Applications (ISDA), 2010 10th International Conference on Nov. 29 2010-Dec. 1, 2010.

[7] Z. Wanlei, Keynote: Detection of and Defense Against Distributed Denial-of-Service (DDoS) Attacks, Proc. IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, 2012.

[8] O. Al-Jarrah and A. Arafat , "Network Intrusion Detection System using attack behavior classification" , 2014 5th International Conference on Information and Communication Systems (ICICS) , pp.1 -6 .

[9] Amrita Big Data Framework [online.available:http://abdf.in/].

[10] Beniwal, Sunita, and Jitender Arora. "Classification and feature selection techniques in data mining." International Journal of Engineering Research and Technology. Vol. 1. No. 6 (August-2012). ESRSA Publications, 2012.

[11] Yanjie Zhao, *et al.* "Network Intrusion Detection System Model Based on Data Mining", Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2016.

[12] Padmashani, R., Shiju Sathyadevan, and Devi Dath. "BSnort IPS better Snort intrusion detection/prevention system." Intelligent Systems Design and Applications (ISDA), 2012 12th International Conference on. IEEE, 2012.

[13] 2dd [online. available: http://2dd.it/].