PERFORMANCE COMPARISON OF SNORT AND SURICATAAND IMPLEMENTATION OF MACHINELEARNING MODEL TO SNORT

TEAM NAME: TROJAN HORSES

TEAM MEMBERS:

ARCHIT GUPTA	21111015
GAJENDER SHARMA	21111028
GARGI SARKAR	21111263
KAJAL SETHI	21111033
PRANSHU SAHIJAWANI	21111048
UTKARSH SRIVASTAVA	21111063

Keywords of the project: Intrusion Detection, Snort and Suricata, Performance Comparison, Machine Learning, Support Vector Machine, Decision Tree

EXECUTIVE SUMMARY OF THE PROJECT

GOAL: The purpose of this project was to compare two intrusion detection systems, Snort and Suricata, and determine which one performed better on various parameters such as CPU utilization, memory utilization, packet drop rate, and the ability to distinguish malicious traffic from legitimate traffic. Due to the high rate of false positive alarms generated by IDS, the final objective was to implement a machine learning model to improve the performance of the higher performing IDS. Prior to integrating machine learning techniques with IDS, we wanted to gain an understanding of high-performing machine learning algorithms.

STAGES: The project is divided into three experimental stages.

In stage 1 two opensource NIDS namely Snort and Suricata were compared. The comparison could be done as both have comparable functions, detection rule sets and syntax. They both support intrusion prevention system (IPS) feature and support medium to high-speed network, Both Snort and Suricata have similar features such as a module to capture the network packets, a module to decode and classify the network packets and a module to detect accurately the malicious or legitimate packets based on a rule set defined by both IDS .To observe CPU utilization, memory utilization and packet drop rate , each IDS was separately installed in Identical VMs with default performance parameter and rule set and later legitimate traffic was generated using the tool OSTINATO. From OSTINATO, 9,00,00,000 UDP packets with rate of 50,000 packets/sec, 9,00,00,000 TCP packets with rate of 50,000 packets/sec, 9,00,00,000 ICMP packets with rate of 50,000 packets/sec were sent.

To determine how accurately Snort and Suricata ruleset inspect network traffic to correctly classify legitimate and malicious traffic, from Sparta, Kali Linux Metasploit framework and NMAP malicious traffic were generated along with the legitimate traffic. then those malicious traffics were injected to both IDS and FPR, TPR were calculated and compared. In this stage, it was found that Snort showed better detection capabilities than Suricata in terms of accuracy but with more false positive alarms.

In stage 2, five different high-performance machine learning algorithms (support vector machine, NaiveBayes, BayesNet, Decision Tree, and Fuzzy Logic) were applied to three standard IDS datasets (NSA Snort IDS Alert Logs, DARPA IDS Dataset, and NSL-KDD IDS Dataset) to determine which algorithm performs the best at reducing false positives. We used the opensource data mining software WEKA in this scenario. The respective FPR, DR, and DA were calculated using MATLAB and the confusion matrix provided by WEKA. We performed tenfold cross validation. If an algorithm's DR is low despite having an excellent FPR or DA, it is rejected. It was observed that Decision Trees and SVMs performed better in this case.

Stage3 is based on Improving rule-based Snort's accuracy and reduce snort's false positive rate with the help of machine learning. As derived from stage 2, a Snort plug-in with SVM and decision tree is proposed. It operates in parallel with the snort rule set .Snort rule set only detects the known malicious traffic, but using this plugin ,the unknown or variant malicious traffic can be detected which in turn will reduce false positive alarm .The idea is to modify the existing snort architecture and integrate the additional plug-in which results in a new architecture. (1) **Decode Network Packets:** This component decodes the packet data to obtain detailed information: source and destination IP addresses/ports, MAC addresses, Ethernet frame and packet size. (2) **Classify Network Packets:** This differentiates between legitimate and malicious traffic. (3) **Machine Learning Algorithm:** Adaptive Plug-in used SVM, Fuzzy Logic, Decision Tree, hybrid of SVM and Fuzzy Logic, and optimized SVM with firefly algorithm to process the legitimate and malicious traffic. (4) **False Positive Alarm Reduction:** This component further reduced the false positive alarms to send the true positive alarms to Snort log files. To select the superior MLA: SVM, Fuzzy Logic or Decision Tree, a live background malicious traffic experiment is done for the Snort adaptive plug-in to evaluate the false positive and false negative alarms rates.

RESULTS

Experiment stage 1 resulted that Suricata's CPU utilization was higher than Snort in the whole procedure.Suricata's average CPU utilization was 25.8359% and Snort's
average CPU utilization was 3.6186%.Suricata's memory usage is greater than that of snort i.e., snort used an average memory of 3.7489% and for Suricata it was 7.3946%.
On the other hand, that Snort showed better detection capabilities than Suricata in terms of accuracy but with more false positive alarms.

METRIC	SNORT	SURICATA
FPR	0.195	0.177
TPR	0.869	0.746
Correctly Classified Instances	86.8774%	74.5785%
Incorrectly Classified Instances	13.1226%	25.4215%

Experiment Stage 2 resulted that Decision Tree (DT) and Support Vector Machine (SVM) performed better in reducing false positive alarms of different IDS datasets.

DATASET: NSL-KDD IDS				DATASET: NSA SNORT IDS				DATASET: DARPA IDS			
MLA	DR	FPR	DA	MLA	DR	FPA	DA	MLA	DR	FPR	DA
SVM	97.74	6.71	93.71	SVM	99	1	100	SVM	96.41	3.7	99.69
DT	97.21	2.54	97.52	DT	99.08	0.9	100	DT	99.82	2.04	98.00
Fuzzy logic	76.7	10.55	90.46	Fuzzy logic	99.97	0.87	99.12	Fuzzy logic	99.79	2.56	97.50
BayesNet	95.39	5.07	95.18	BayesNet	99.97	0.31	99.68	BayesNet	99.27	7.59	92.94
NaiveBayes	95.00	30.06	76.89	NaiveBayes	99.97	0.75	99.24	NaiveBayes	99.11	10.23	90.72

For Experiment Stage 3, we were able to capture real time Snort Logs, parse them, and also able to predict if they were malicious or not. However, due to bias in the dataset, the model got overfit and couldn't predict accurately. Had it worked properly, the final step would have been filtering the 'malicious' logs as predicted by the classifier, in order to give the filtered Snort logs, with low FPR.

CONCLUSION:

- We were able to clean and analyse popular Network Datasets such as NSA Snort, DARPA, KDD.
- Also we compared 5 Machine Learning Classification Algorithms and figured out the best one based on the above datasets
- We aimed at building a plugin in order to reduce the FPR of Snort. However due to bias in the dataset(manually captured logs), we couldn't complete the classification task, accurately. This can be improved by doing different kinds of attacks and experimenting with more powerful rulesets, in order to build an unbiased dataset.