STANDARDISATION AND CLASSIFICATION OF ALERTS GENERATED BY INTRUSION DETECTION SYSTEMS

Athira A B[1] and Vinod Pathari[2]

[1]Department of Computer Engineering ,National Institute Of Technology Calicut, India
[2]Department of Computer Engineering ,National Institute Of Technology Calicut, India

ABSTRACT

Intrusion detection systems are most popular de-fence mechanisms used to provide security to IT infrastructures. Organisation need best performance, so it uses multiple IDSs from different vendors. Different vendors are using different formats and protocols. Difficulty imposed by this is the generation of several false alarms. Major part of this work concentrates on the collection of alerts from different intrusion detection systems to represent them in IDMEF(Intrusion Detection Message Exchange Format) format. Alerts were collected from intrusion detection systems like snort, ossec, suricata etc. Later classification is attempted using machine learning technique, which helps to mitigate generation of false positives.

KEYWORDS

Intrusion Detection Systems, IDMEF, Snort, Suricata, ossec& WEKA

1. INTRODUCTION

Due to the widespread use of Internet, providing security against attacks on network is a challenging job today. Most of the organisations use intrusion detection systems (IDS) for providing security. Need for IDS can be summed up as simple principle of security: Defence in Depth. It is a layered approach involving multiple overlapping controls in preventing, detecting and responding to suspected intrusions.

1.1. INTRUSION DETECTION SYSTEM

Intrusion detection systems are most popular defence mechanisms used to provide security to IT infrastructures. Intrusion is a sequence of related actions performed by a suspicious adversary, which result in the form of compromise of a target system [7]. These kinds of actions violates certain security policy of the system. The process of identifying and responding to suspicious activities of target system is called Intrusion Detection [7].

DOI: 10.5121/ijci.2016.5203

1.2. MOTIVATION

Organisations frequently use several IDSs from different vendors since each has its relative strengths. One may be strong at host-based intrusion detection while another may be strong at network based intrusion detection. Organisations need best performance, do not prefer to take a chance with security and hence use multiple IDSs from different vendors. Different IDSs will be using different protocols and generate alert events in different formats. If we fail to integrate the outputs from all these properly, the volume of data generated will be high and accordingly more false positives occur. Large volume of IDS false alarms is unacceptable to security administrators as it hinders smooth functioning of any organization. To reduce the cost of operation and increase the reliability of a security system, it is required to tackle the excess of false alarms.

2. PROBLEM STATEMENT

To develop an approach to collect alerts from different sensors and standardize them into IDMEF. Later these alerts will be classified into false alarms and attacks attempted using machine learning technique.

3. RELATED WORKS

KleberStoreh et al. [9] proposed an approach for correlat-ing security events using machine learning technique. Layered approach is followed here. Apart from normal methods they analyse alerts generated from different sensors, which are normalised, fused into meta-alerts and are then used for classification into alerts or false alarms. ChampaDey [7] proposed a similar approach for reducing false alarms using incremental clustering algorithm. Only data from snort IDS is used for analysing purpose. The alert data is then processed using incremental clustering algorithm and classified into alerts or false alarms.

4. PROPOSED METHOD

In the proposed system, format difference in alert from different sensors is overcome by representing them into IDMEF (Intrusion Detection Message Exchange Format) format. Later classification of parsed IDMEF alerts into false alarms and attacks is achieved using machine learning technique. In this work, we collect alerts from different intrusion detection systems and proceed as follows:

- Convert collected alerts into a common format (ID-MEF is identified as common format).
- Labelling of alerts.
- Classification of alerts into false alarm or attack using machine learning technique.

Detailed work flow for the proposed system is shown in Figure 1.

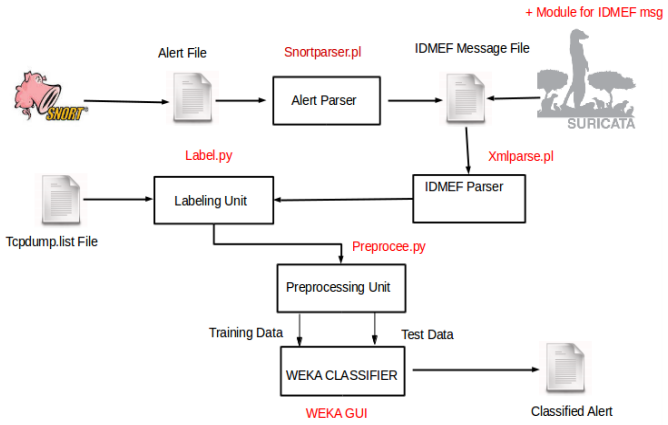

Figure 1. Detailed Work Flow

5. STANDARDISATION OF ALERTS

5.1. IDMEF

IDMEF(Intrusion Detection Message Exchange Format) is an object oriented representation of alert data generated by intrusion detection systems. The goal of IDMEF is a standard representation of alert data in an unambiguous manner. IDMEF data model can be summarised as Figure 3.1 [11] .Two types of implementation for IDMEF was proposed by Intrusion Detection Working Group (IDWG) [11]. One method is using Structure of Management Information (SMI) [11] and the other is using XML. During second phase of our work, we need to process the IDMEF messages. Software tools for processing XML documents are widely available, in both commercial and open source forms [11]. Hence we chose to implement IDMEF in XML format.

5.2. IDMEF GENERATION

DARPA (Defence Advanced Research Project Agency) [1] data sets are used for testing. DARPA simulate American air force based local network being attacked in different ways. Attack information are provided in the form of log files. DARPA data set is replayed using different IDSs.We considered Snort and Suricata IDSs. Alerts were gathered from them and IDMEF messages were generated. IDMEF message generation details are explained in the following sections.

5.2.1. Snort

Snort is a widely used open source signature based network intrusion detection system, configured to operate on Network IDS mode. In Network IDS mode, snort will perform actual analysis to determine malicious traffic and alerts are generated. To conduct testing DARPA 1998 data sets were downloaded from MIT Lincoln Labs website [1]. This dataset contains simulated network traffic embedded with marked attacks. snort was configured in network intrusion detection system to use this data set. We wrote a perl script to attain the task of standardisation phase in work flow diagram. Alert file serves as input to this program. Required alert attributes

are obtained through parsing and IDMEF message is obtained with the help of XML::IDMEF library. The IDMEF messages obtained from snort alert file is shown in Figure 2.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<idmef:IDMEF_Message xmlns:idmef="http://iana.org/idmef version =1.0">
 <idmef:Alert messageid="1">
  <idmef:Analyzer analyzerid="527">
   <idmef:Node category="8" />
  </idmef:Analyzer>
 </idmef:Alert>
 <idmef:CreateTime ntpstamp="06/16-11:22:28.515219" />
 <idmef:Source>
  <idmef:Node>
   <idmef:Address category="ipv4-addr">
    <idmef:address>0.0.0.0</idmef:address>
   </idmef:Address>
  </idmef:Node>
  <idmef:Service>
   <idmef:priority> 2</idmef:priority>
   <idmef:protocol>IGMP</idmef:protocol>
  </idmef:Service>
 </idmef:Source>
 <idmef:Target>
  <idmef:Node>
   <idmef:Address category="ipv4-addr">
    <idmef:address>224.0.0.22</idmef:address>
   </idmef:Address>
  </idmef:Node>
  <idmef:Service>
   <idmef:protocol>IGMP</idmef:protocol>
  </idmef:Service>
 </idmef:Target>
 <idmef:Classification text=" Potentially Bad Traffic" />
</idmef:IDMEF_Message>
```

Figure 2. IDMEF Message Generated by Snort

5.2.2. Suricata

Suricata, a rule-based IDS, take advantage of the externally developed rule sets to monitor sniffed network traffic and provide alerts when suspicious events take place. Suricata uses the Yaml format for configuration. suricata.yaml file included in source code is the example configuration file of Suricata. After packet analysis Suricata generates alert outputs. Output section in suricata.yaml controls the output structure for alerts generated. Default log directory is /var/log/suricata. There are several types of output structures like fast.log, http.log, stats.log etc. To generate IDMEF messages an output structure as mentioned above was developed. For this we have developed a C program, which will write data into buffer in IDMEF format. Program files were appended to source code. Re-installation of Suricata was performed. Suricata was configured to use DARPA data set. Alerts were generated from the suricata. IDMEF messages are generated at default directory /var/log/suricata/fast.log as shown in Figure 3.

```
<?xml version="1.0" encoding="UTF-8"?>
<idmef:IDMEF-Message version="1.0"xmlns:idmef= http://iana.org/idmef/>
<idmef:Alert messageid= 2200075 >
<idmef:Analyzer analyzerid=1>
</idmef:Analyzer>
<idmef:CreateTimentpstamp=03/12/2015-15:36:31.734831 >
<idmef:Source>
<idmef:Node>
<idmef:Addresscategory=ipv4-addr>
<idmef:address>192.168.4.71</idmef:address>
</idmef:Address>
</idmef:Node>
<idmef:Service>
<idmef:priority>3</idmef:priority>
<idmef:protocol>UDP</idmef:protocol><idmef:port>65419</idmef:port>
</idmef:service>
</idmef:source>
<idmef:Target
<idmef:Node>
<idmef:Addresscategory=ipv4-addr>
<idmef:address>192.168.254.2</idmef:address>
</idmef:Address>
</idmef:Node>
</idmef:Target>
<idmef:Service>
<idmef:protocol>UDP</idmef:protocol>
<idmef:port>53</idmef:port>
</idmef:service>
</idmef:Target>
<idmef:Classificationtext=(null)>
</idmef:Classification>
</idmef:Alert>
</idmef:IDMEF-Message>
```

Figure 3. IDMEF Message Generated by Suricata

6. FALSE POSITIVE REDUCTION

6.1. ALERT CLASSIFICATION

As we discussed earlier the main objective of intrusion detection system is to distinguish between attacks and normal events. Most of intrusion detection systems face a common problem which is the generation of high false alarms. An IDS is efficient when it contains less number of false positives and false negatives. One way to tackle this problem is using machine learning technique. Machine learning techniques can be used to distinguish between attacks and false alarms.

a. MACHINE LEARNING

DARPA data set provide tcpdump.list files. For each online traffic, information about attacks in each connection will be included in tcpdump.list files. Connection is a sequence of TCP packets starting and ending at some well defined time interval. Between this connections data flow from one source IP address to target IP address under the control of a protocol. Input to labelling unit are two files, alertlog file and tcpdump.list file. tcpdump.list file contain information about start date, duration, service, source port, destination port, source IP, destination IP, attack score and

attack type. Attack score is a binary valued attribute. Presence of an attack is indicated by an attack score 1 while 0 indicates the absence of an attack. Attacks are mainly divided into five classes DOS, Probe, R2L, U2R, DATA . Algorithm for Labelling Alerts The algorithm is implemented in python. The labelled alert file is used for classification. Classification is attempted using machine learning algorithm. We use WEKA tool for this approach.

Input: Tcpdump.list File, Alertlog (parsed IDMEF file) File
Output: Labelled Alerts

1. For each row in tcpdump.list files
   If row is a labelled attack then add the row to the new file AttackList
2. For each row in alertlog file
   Create key with three attributes timestamp, srcip, destip
   IF
   The key exists in the AttackList file, Identify the attack class for the type of attack found. Label the selected row with the type of attack class.
   Else
   Label the selected row as normal
3. Return the AlertList file

Algorithm 1: Algorithm for Labelling Alerts

6.3. WEKA

Weka(Waikato Environment for Knowledge Analysis) [5] is a free and open source tool used for data mining tasks. Weka has many applications like Explorer, Experimenter, Knowledge Flow and Simple CLI. We attempt classification using Weka Knowledge Explorer.

6.4. WEKA EXPLORER

The classifier panel in Weka Explorer allows us to configure and execute any weka classifier on the current data set. We take data set with known output values and use this to build a data model. Whenever we have new data points with unknown output values, we put it through model and produce our expected output. This model requires one extra step, shown as pre-processing unit in Detailed work flow diagram in Figure 1. Entire training set will be taken and divided into two parts. We will take about 60-80 % data and put into our training set, which will be used to create the data model. Then take the remaining data and use it as test set, which will be used for testing the accuracy of our model after creating it. A Naive Bayesian Learner (bayes.Naive Bayes) algorithm will be used for classification.

6.5. RESULTS AFTER CLASSIFICATION

One way to evaluate IDS is by its prediction ability to give a correct classification of events to be attacks or normal behaviour. According to real nature of an event the prediction from an IDS has four possible outcome which is called confusion matrix.

Table1. Confusion Matrix

	Normal	**Attack**
Normal	TN	FP
Attack	FN	TP

- TN :-True Negatives are actually normal events and successfully labelled normal.
- TP:- True Positives are attack events and successfully labelled as attacks.
- FP:- False Positives are normal events being classified as attacks.
- FN:- False Negatives include attack events incorrectly classified as normal events.

True negatives and True positives corresponds to the correct operation of the IDS.

False Positive Rate(F P R) = FP/FP+TN (1)

Also known as false alarm rate. Rate at which normal data will be falsely detected as attacks. High FPR will degrade the performance of IDS.

False Negative Rate(F NR) = FN/TP+FN (2)

If FNR is high system is vulnerable to attacks.

True Positive Rate(T P R) = TP/TP + FN (3)

True Negative Rate(T N R) = TN/TN + FP (4)

Also known as detection rate or sensitivity. It is the ratio of detected attacks among all attack events.

Accuracy =TP + TN/TP + TN + FP + FN (5)

It is the ratio of events classified as accurate type in total events.

Precision = TP/TP+FP (6)

Figure 4 shows result after classification.

```
Time taken to build model: 0.04 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances        1399               80.2179 %
Incorrectly Classified Instances       345               19.7821 %
Kappa statistic                          0.0484
Mean absolute error                      0.133
Root mean squared error                  0.2555
Relative absolute error                106.4934 %
Root relative squared error            102.2921 %
Total Number of Instances             1744

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0         0.005      0         0         0          0.294    dos
                 0.975     0.911      0.83      0.975     0.896      0.765    normal
                 0.031     0.019      0.143     0.031     0.051      0.245    probe
                 0         0.012      0         0         0          0.002    R2L
                 0         0          0         0         0          0.304    U2R
Weighted Avg.    0.802     0.749      0.694     0.802     0.74       0.675

=== Confusion Matrix ===

    a    b    c    d    e   <-- classified as
    0  130   16    2    0 |    a = dos
    4 1394   14   18    0 |    b = normal
    4  151    5    1    0 |    c = probe
    0    1    0    0    0 |    d = R2L
    0    4    0    0    0 |    e = U2R
```

Figure 4. Result After Classification

3. CONCLUSIONS

Organisations frequently use several IDS from different vendors since each have relative strengths and weaknesses. The use of diverse IDS solution leads to generation of too many false positives. If we fail to tackle the problem it will effect the performance of organisations. In the proposed system, format difference in alert from different IDSs are overcome by representing them into IDMEF format. Alert data can be handled efficiently by representing alerts into IDMEF message. Later classification of parsed IDMEF alerts into false alarms and attacks is achieved using machine learning technique. Parameters obtained by parsing IDMEF were not optimised in our approach. This will further improve the performance of alert classification.

ACKNOWLEDGEMENTS

We would like to show our gratitude to everyone for sharing their pearls of wisdom with us during the course of this research, and who provided insight and expertise that greatly assisted the research.

REFERENCES

[1] DARPAdataset, http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/data/. Accessed on 03-December-2014.

[2] Ossec, http://www.ossec.net//. Accessed on 03-December-2014.

[3] Snort, https://www.snort.org/. Accessed on 03-December-2014.

[4] Suricata, https://redmine.openinfosecfoundation.org/projects/suricata/ wiki/Suricatayaml. Accessed on 2-February-2015.

[5] HadiBahrbegi Mir Kamal Mirnia Mehdi BahrbegiElnazSafarzadeh Amir AzimiAlastiAhrabi, Ahmad HabibizadNavin and Ali Ebrahimi, "A New System for Clustering and Classification of Intrusion Detection System Alerts Using Self-Organizing Maps", International Journal of Computer Science and Security, 4, 2004.

[6] Neethu B, "Classification of Intrusion Detection Dataset using machine learning Approaches", International Journal of Electronics and Com-puter Science Engineering, 1956.

[7] ChampaDey, "Reducing ids false positives using Incremental Stream Clustering (isc) Algorithm", Dept of Computer and Systems Sci-ences, Royal Institute of Technology, Sweden, page March, JULY-SEPTEMBER 2009.

[8] Debar H and Wespi A, "Aggregration and Correlation of Intrusion-Detection Alerts", In Proceedings of the 4th International Symposium on Recent Advances in Intrusion detection (RAID), Springer Verlang, California, USA, pages 85–103, 2001.

[9] KleberStroeh, Edmundo Roberto Mauro Madeira, and Siome Klein Goldenstein, "An approach to the correlation of security events based on machine learning techniques", Journal of Internet Services and Applications, 2013.

[10] SebastiaanTesink, "Improving intrusion detection systems through machine learning", ILK Research Group,Technical Report Series no. 07-02, Tilburg University, page March, JULY-SEPTEMBER 2007.

[11] FredrikValeur, Giovanni Vigna, and Christopher Krue, "Modeling In-trusion Alerts using idmef", IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, 1(3), JULY-SEPTEMBER 2004.

Authors

Athira A B- She received the B.Tech. Degree in computer science and engineering from University of Calicut, Kerala, India, in 2012, and M.Tech.in computer science and engineering (Information Security) from the National Institute of Technology (NIT) Calicut, Kerala, India in 2015.

VinodPathari- He is working as a full time faculty in the Computer Science and Engineering Department of NIT Calicut, Kerala, India. In addition to information security related topics he is also interested in teaching functional programming and software engineering.