Journal of Theoretical and Applied Information Technology
20th January 2016. Vol.83. No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A NOVEL FILTER BASED PARTITIONING DECISION TREE MODEL FOR REAL-TIME NETWORK SECURITY

[1] SITA RAMA MURTY PILLA, [2] R KIRAN KUMAR, [3] M SAILAJA

[1]Departement of IT, Vishnu Institute of Technology, Bhimavaram, Andhra Pradesh, INDIA

[2]Departement of CSE, Krishna University, Machilipatnam, Andhra Pradesh, INDIA

[3]Departement of ECE, JNTUK University, Kakinada, Andhra Pradesh, INDIA

E-mail: [1]psramam.1@gmail.com, [2]kirankreddi@gmail.com

ABSTRACT

Due to the exponential rise of the network attacks and increasing development of software tools and techniques for intrusion detection, the rule based intrusion detection system has become an essential solution for real-time anomaly detection. Basically, traditional data mining based intrusion detection methods generate a large set of predefined patterns most of them are high false rate and inaccurate. There is a need to optimize the real-time network attacks due to the variation in new attack type, instance set and attributes. To address the issue of high false rate and dynamic data integration, a new anomaly detection system using data mining model has been proposed to find the real time DOS/DDOS patterns by integrating network packets capturing from different systems on the network and kdd99 dataset. This system generates intrusion patterns by integrating the predefined attacks and new attacks as early as possible with low false rate. Experimental results show that proposed dynamic model optimizes the real-time true positive patterns with high accuracy compared to traditional models.

Keywords: *DDOS, Attribute Selection, Decision Tree, Intrusion Detection, KDDCup 99 Intrusion dataset*

1. INTRODUCTION

In recent years, computers and the internet have been utilized by many network analysts all over the world in different platforms. Intrusion detection system provides an essential role to secure the network environment by using different rules or patterns. Yet, the intrusion detection system adopted the conventional data mining models to generate the intrusion patterns from different types of networks have limited validity and scalability. Some signatures poorly describe the attack, making them trigger on benign traffic as a result. Instead of using regular expressions to describe an attack, simple string matching is often used. This has several reasons; due to processing time restrictions simpler string-matching is used to be able to keep up with traffic data. Writing good and correct signatures is a difficult task and often leads to buggy or incomplete signatures. Some signatures trigger on rare or suspicious traffic. These in themselves are not classified as attacks, but are considered uncommon, i.e. failed logins, overlapping IP fragments or the use of the URGENT bit in the TCP header [1-3]. It has been shown that these alerts often are not linked to malicious activities. In a network environment, there are three types of network intrusion systems exist, they are- network based intrusion detection systems, distributed intrusion detection system and host based intrusion detection systems. Anomaly learns what is considered a normal traffic on a system and from this model alerts on any traffic deviating from this. Misuse detection on the other hand, uses search for fixed patterns within the traffic it knows is malicious. According to the different network technologies and platforms, network intrusion systems can be categorized in two ways: signature based and anomaly based detection. There are different factors about models and data mining algorithms in intrusion detection.

- Selecting and building sequential and association analysis.
- Building self detection and self aware learning rules such as clustering and pattern analysis.
- Traditional machine learning models are time consuming in attack evaluation, if the data size is exponentially increased. So, it

is difficult to find the attacks in real time manner.

With the deployment of IDS one of their weaknesses is becoming visible, false positives. False positives are alerting from an IDS on non malicious activity. With the increase in worm activity and more complex network structures the amount of alerts from IDSs have increased, making it virtually impossible to check the validity of every. To reduce these problems different data mining approaches have been used. Data mining is an algorithm developed to search through a vast amount of data looking for patterns or relations, which can prove to be useful. These algorithms are often time consuming and require a lot of time to complete, but with the ever increasing processing power they are becoming more and more applicable. The patterns found are often surprising, and can provide further insight [4].

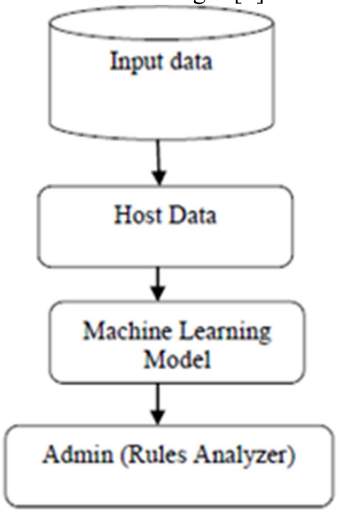


Figure 1: Traditional based NID Model

Responding to and evaluating an IDS alert is a laborious intensive task, requiring vast human resource. With the increasing amount of alerts the task of handling these alerts are becoming a daunting task. Many system administrators often ignore or refuse to deploy intrusion detection systems due to their known high rate of alerts, missing out on many of the great benefits an intrusion detection system can provide in a network. The base framework of data mining methodology of intrusion detection system is shown in Fig 1. Initially, data are collected from the connected network as data stream and generates specific packets as records. After that, machine learning algorithms are used to learn and analyze the input data to find the normal and abnormal patterns. Intelligent decision making models decide whether the intrusion takes place or not using the generated patterns from the data mining model to the administrator.

Research Objectives:

- An integrated model for real time attack classification using an offline attack database.
- Real-time attack filtering model for mixed data type.
- Multi-class decision tree construction using probabilistic measures.

2. RELATED WORK

[2][3] have implemented the need for a framework and architecture specification for intrusion detection systems. But the details of the framework and infrastructure required to support the complex datasets are not included, instead they handled data with a limited number of attributes and restricted constraints. Several methods have been implemented recently to find the correlation between intrusion detection alarms. Probabilistic based alert correlation finds the similarity between alerts that match closely, if not exactly.

The classification model can be decision tree based, rule based, Bayesian network based, association rule based and neural network based IDS. These models ensure that no intrusion will be missed while checking the real time attack on the network [4]. Anomaly detection algorithms which identify new kind of attacks based on deviations from the regular usage or patterns [4,5]. In statistics based intrusion detection , the data objects are modeled using outlier detection depends on the feature relationship. However, as the input data size increasing, it becomes difficult to process and in accurate to predict the data distribution of the data objects.

[5] implemented a classification model for intrusion detection that can be achieved as, different methods like linear discrimination, KNN are used to scan the network traces.[6] implemented a genetic technique and used a binary decision tree to represent the data. They used the false positive rate and detection rate as condition criterion among the dataset.[7] also implemented a genetic method for sparse trees to find anomalies.

[8] has implemented naïve Bayesian model based decision tree to form the patterns which are redundant and less informative. Chavan [9] used a decision tree model for extracting relevant features through ranking approach per each class. They extracted 41 features to 16 features for normal type and 13 features for attack detection.

[10] Developed comparative analysis of decision tree model vs naïve Bayesian algorithms and generates combined intrusion rule for decision making. Ensemble techniques have the major benefits as they can be applied to make feature changes in the input data stream more effectively than traditional framework models. In this model, after the ith decision-tree construction, the total classification error rate of the decision tree is defined as the commutative sum of the instance weights which are not a true positive over the commutative sum of all input instance weights, and is computed as:

$$err_k = \sum_{m(falseinst)} w^k{}_m / \sum_m w^k_m$$

Where m=1, 2…….. instances of the input data set Machine learning approaches like classification and clustering of malware have been proposed on reports generated from dynamic analysis. Models are built for malware families whose labels are available and used for predicting the malware labels for newly seen sample reports. Most malware datasets are collected over a certain period of time using a honeynet setup and they comprise executables, aimed to attack Window based systems. The dynamic analysis techniques gained prominence because of the limitations in the static analysis techniques.[7-9] proposed a method where the normal model of programs were modeled using sequences of system calls and any deviations from this was aged as an anomaly or a security threat. This was one of the first approaches of using behavior to differentiate malware from benign programs.

3. PROPOSED MODEL

In this model, data captured from the realtime networks like LAN/WAN are saved in a database for filtering. In the field extraction phase, relevant fields corresponding to the attack are extracted to improve the true positive of the attack. For instance, the fields which are relevant to DOS/DDOS type of attack are IP, Src_bytes, packet length, duration, interface, header flags, src_address, dest_address etc. After the field extraction phase, the instances which are relevant to a specific type of attack are identified and are being saved in DB. This process is repeated until time_interval is satisfied or the size of the instances is reached. This attacked dataset and the traditional kddcup'99 dataset are integrated to form a new dataset for data mining framework. Proposed filtered based partitioning decision tree can be used to generate attacked patterns on the new dataset with high true positive rate compared to traditional models

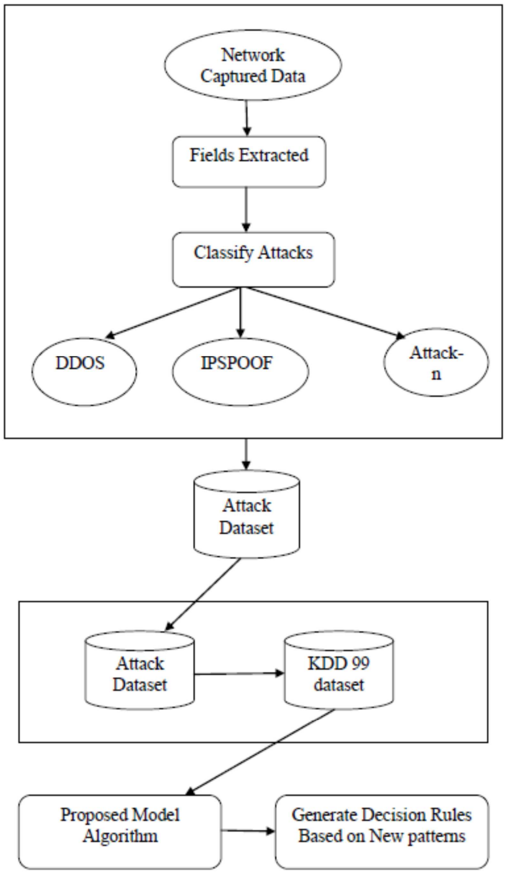


Figure 2: Proposed Model Overview

3.1 Network Field Extractor Algorithm

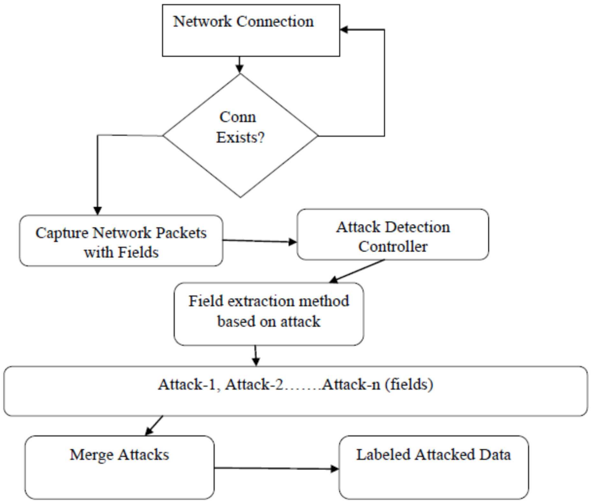

Network Connection
Conn Exists?
Capture Network Packets with Fields
Attack Detection Controller
Field extraction method based on attack
Attack-1, Attack-2…….Attack-n (fields)
Merge Attacks
Labeled Attacked Data

Figure 3: Workflow Of Packet Filtering

Field Extraction algorithm for DDOS:
Input: Packets with Network fields, Boolean flg=false;
Output: Attack Fields with labels.
Procedure:

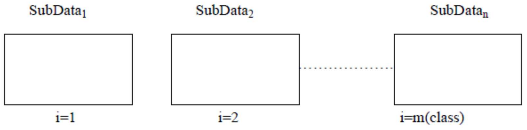
```
Netconn[]=getInterface(LAN/WAN);
For each interface do
If(Netconn[i]>0)
Then
Connect=true;
Interconn=Netconn[i](connect true);
End if
End for
If flg==false
Then
Disconnect
End if
AttackField[i]=NULL;
PacketFields[]=NULL;
For each captured packet
Do
Packet[]=getPacket(Interconn);
Done.
For each packet $pk_i$ do
$L_i = getLength(\mathrm{pk}_i)$
$if\ \mathrm{L}_i > thresh(\sigma)$
$then$
$attack(\mathrm{C}_m, \mathrm{L}_i) = pk_i;$
$endif$
$f\ \mathrm{count}(\mathrm{pk}_i) > \phi$
$then$
$attack[C_m, Count(\mathrm{pk}_i)] = Count(\mathrm{pk}_i);$
$endif$
$endfor$
$Display$ attacked packets with DOS/DDOS label
$\mathrm{NewD}_i = attack;$
```

In this algorithm, local Ethernet or wireless router, any network interface in the same network type can access the data or receive the data to or from the network interfaces. Each PC in the connected network has fields such as IP address, TCP, UDP, RARP and ARP protocols are used to communicate with each other. In any real time network, there are four different ways to send or receive data such as broadcast, unicast , multicast and hybrid mode. Non-promiscuous mode can take any packet through the network interface with considering whether the data is passed or not. In promiscuous mode, any system can capture packets through the device, without considering whether the data is passed or not. We used Winpcap library as the development tool to support packet capturing and field extraction process. In this detection process, each captured data is checked against the attack using the specified packet size as threshold (σ). If the condition is met, then the packets are saving in a file and labeled as DOS/DDOS attack class name.

3.2 Algorithm 2: Data Integration and Filtering

Input: NewD_i (New attacked dataset), $\mathrm{OldD}_i(Kdd99)$
Output: Filtering dataset.

```
For each $\mathrm{OldD}_i$
Do
For each $\mathrm{NewD}_i$
Do
If ( $mClass(\mathrm{Old\,D}_i) == mClass(\mathrm{NewD}_i)$ )
Then
Map(m, $\mathrm{OldD}_i$ , $\mathrm{NewD}_i$ );
End if
End for
End for
```



```
i=m(class)
// Get DOS or DDOS class instances from the mapped list based on protocol, srcbytes, duration and class type.
For each class value $m_i$ in Map(m, $\mathrm{OldD}_i$ , $\mathrm{NewD}_i$ )
Do
if (Type( $m_i$ )==DOS || Type( $m_i$ )==DDOS)
then
List $v_o = Get(m_i, OldD_i)$ ;
List $v_N = Get(m_i, \mathrm{New}\ D_i)$
If(correlation( $(v_O, v_N)$ )>0.5)
Then
NewKDD $(v_O, v_N)$ ;
End if
End if
```

```
Done
$D_{ij} = NewKDD(v_O, v_N);$
```

Algorithm 2 describes the relationship between the instances in the data integration phase. In this phase, newly captured instances correlate with the traditional kdd data for data integration. If the correlation between the old and new instances is greater than predefined threshold 0.5, then those instances are added to the database for classification.

3.3 Algorithm 3: Proposed Decision Pattern Miner

```
Input: $D_{ij}$ (integrated dataset)
Output: Decision patterns
Procedure:
If $D_{ij}$ ==Null
Then
    Return leaf node with attack pattern empty.
Else if class ( $D_{ij}$ ) ==1
Then
    Return leaf node with attack m.
Else
    Split $D_{ij}$ into k disjoint partitions using stratified random sampling distribution where k=m(classes).
    Let $D_1(i1,j1), D_2(i2,j2)...D_k(ik,jk)$ are k disjoint partitions with m classes such that
    $D_{ij} = D_1(i1,j1) \cup D_2(i2,j2)... \cup D_k(ik,jk)$
    Let $A_1(n), A_2(n)...A_k(n)$ corresponds to n attributes of the $D_1(i1,j1), D_2(i2,j2)...D_k(ik,jk)$ partitions.
    i.e $A_1(n)$ corresponds to the attribute list of the data partition $A_1(n)$.
    For each partition do
        Find the attribute ranking using the following equation
        AttRank(P, $A_i(n)$)=
        $\{\sum prob(A_i(i)/A_j(i))\}_{m=1.2..classes} / \sum_m prob(A_i(j))$
        Where i,j=1,2….n attributes and m=number of classes.
        $prob(A_i(n))$ : probability of the tuples satisfying
        If m==DOS/DDOS and AttRank(P, $A_i(n)$ ,m)<0.5
        Then
            AttRank(P, $A_i(n)$ )= AttRank(P, $A_i(n)$ )+0.5;
        End if
    End for
    Select the root node in the tree pattern using the attribute with highest AttRank in all the partitions.
    This process is repeated until no more instances in the partitions.
Display attacked patterns in the decision tree.
```

tion has to be in sentense case with no spacing above or below the start of it.

4. EXPERIMENTAL RESULTS

In this section different attack patterns are analyzed using real-time network data and KDD'99 dataset. Experimental results are simulated using the Java programming environment with third party libraries such as JAMA, JUNIT, and statistic and pattern miner. In our experiment, we have captured network packets using LAN/WAN and then attack detection operation was initiated. Experimental results prove that proposed pattern detection model outperforms well compared to traditional static models.

Proposed Attack Patterns

```
service = http
| same_srv_rate < 0.82
| | src_bytes < 89.5 ==> DDOS
| | src_bytes >= 89.5 ==> normal
| same_srv_rate >= 0.82
| | src_bytes < 28483
| | | dst_host_srv_serror_rate < 0.67
| | | | src_bytes < 256.5
| | | | | src_bytes < 206.5
| | | | | | dst_bytes < 2463
| | | | | | | src_bytes < 203.5
| | | | | | | | srv_diff_host_rate < 0.09
| | | | | | | | | dst_host_srv_count < 219.5
| | | | | | | | | | dst_host_srv_count < 214.5 ==> normal
| | | | | | | | | | dst_host_srv_count >= 214.5 ==> back
| | | | | | | | | dst_host_srv_count >= 219.5 ==> normal
| | | | | | | | srv_diff_host_rate >= 0.09
| | | | | | | | | srv_count < 33
| | | | | | | | | | dst_host_srv_diff_host_rate < 0.36
| | | | | | | | | | | src_bytes < 199.5 ==> normal
| | | | | | | | | | | src_bytes >= 199.5
| | | | | | | | | | | | dst_host_same_src_port_rate < 0.01
| | | | | | | | | | | | | dst_bytes < 1710.5 ==> back
| | | | | | | | | | | | | dst_bytes >= 1710.5 ==> normal
| | | | | | | | | | | | dst_host_same_src_port_rate >= 0.01 ==> normal
| | | | | | | | | | dst_host_srv_diff_host_rate >= 0.36 ==> back
| | | | | | | | | srv_count >= 33 ==> back
| | | | | | | src_bytes >= 203.5
| | | | | | | | srv_diff_host_rate < 0.42 ==> normal
| | | | | | | | srv_diff_host_rate >= 0.42
| | | | | | | | | src_bytes < 204.5 ==> teardrop
| | | | | | | | | src_bytes >= 204.5 ==> normal
| | | | | | dst_bytes >= 2463
| | | | | | | srv_count < 7.5
| | | | | | | | dst_host_srv_count < 134.5
| | | | | | | | | dst_host_count < 46 ==> normal
| | | | | | | | | dst_host_count >= 46 ==> smurf
| | | | | | | | dst_host_srv_count >= 134.5 ==> normal
| | | | | | | srv_count >= 7.5
| | | | | | | | dst_host_count < 8.5
| | | | | | | | | src_bytes < 199 ==> land
| | | | | | | | | src_bytes >= 199 ==> teardrop
| | | | | | | | dst_host_count >= 8.5
| | | | | | | | | srv_diff_host_rate < 0.26
| | | | | | | | | | dst_bytes < 4819.5
| | | | | | | | | | | dst_bytes < 4793 ==> normal
| | | | | | | | | | | dst_bytes >= 4793 ==> land
| | | | | | | | | | dst_bytes >= 4819.5 ==> normal
| | | | | | | | | srv_diff_host_rate >= 0.26
| | | | | | | | | | dst_bytes < 16590.5 ==> teardrop
| | | | | | | | | | dst_bytes >= 16590.5
| | | | | | | | | | | src_bytes < 152 ==> normal
| | | | | | | | | | | src_bytes >= 152 ==> land
| | | | | src_bytes >= 206.5
| | | | | | srv_diff_host_rate < 0.08
| | | | | | | dst_host_srv_diff_host_rate < 0.01
| | | | | | | | flag = SF
| | | | | | | | | srv_diff_host_rate < 0.04
| | | | | | | | | | dst_bytes < 1482
| | | | | | | | | | | dst_host_same_src_port_rate < 0.01 ==> normal
| | | | | | | | | | | dst_host_same_src_port_rate >= 0.01
| | | | | | | | | | | | count < 9.5 ==> normal
| | | | | | | | | | | | count >= 9.5
| | | | | | | | | | | | | srv_count < 11.5 ==> back
| | | | | | | | | | | | | srv_count >= 11.5 ==> normal
| | | | | | | | | | dst_bytes >= 1482
| | | | | | | | | | | count < 6.5 ==> normal
| | | | | | | | | | | count >= 6.5
| | | | | | | | | | | | dst_bytes < 1483.5 ==> smurf
| | | | | | | | | | | | dst_bytes >= 1483.5
| | | | | | | | | | | | | count < 31.5
| | | | | | | | | | | | | | src_bytes < 249.5 ==> normal
| | | | | | | | | | | | | | src_bytes >= 249.5
| | | | | | | | | | | | | | src_bytes < 250.5
| | | | | | | | | | | | | | | | count < 20.5 ==> smurf
| | | | | | | | | | | | | | | | count >= 20.5 ==> normal
| | | | | | | | | | | | | | | src_bytes >= 250.5 ==> normal
| | | | | | | | | | | | | count >= 31.5
| | | | | | | | | | | | | | dst_bytes < 3424 ==> normal
| | | | | | | | | | | | | | dst_bytes >= 3424 ==> smurf
| | | | | | | | | srv_diff_host_rate >= 0.04

Generated Dynamic Intrusion Rules: 921
Time taken to run Proposed Model: 0.52 seconds
Time taken to detect attacks in the test model on training Model: 0.19 seconds
=== ACCURACY DETAILS ===
Correctly Classified Accuracy        : 0.98150
Total Statistical Error Rate         : 0.22859 %
Real Time Attack Detection Rate      : 0.95249
Total Number of Instances            : 5291
```

Table 1: Captured Packets In Different Experiments Along With Packet Size And Time To Detect The Attack

Sample Packets	DOS/DDOS (Packets)	TimeToDetect (ms)
10000	2466	5877
20000	4803	6388
30000	7988	8655
40000	12866	9899

Table.1 describes the sample captured packets along with DOS attacked information. Time to detect the DOS/DDOS in LAN/WAN networks.

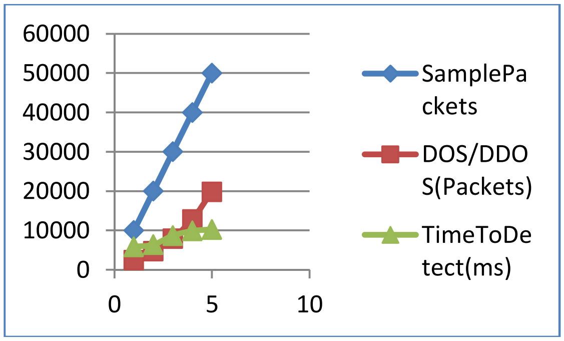


Figure 4: Packet Sizes Vs Detection Times

Table 2: Represents The Integrated Datasize With Detected Attack Patterns Or Rules And Accuracy Measures.

	Threshold=0.5		
Kdddatasize	attack rules	Accuracy	Kdddatasize
4000	834	0.94	0.28
5000	924	0.96	0.18
6000	1025	0.97	0.26
7000	1129	0.985	0.27
8000	1433	0.979	0.22

Table 2, describes the realtime integrated coded dataset along with attacking decision rules. From the table, it is observed that the attack detection accuracy and error rate in the proposed model increases, when the user defined threshold was initialized as 0.5.

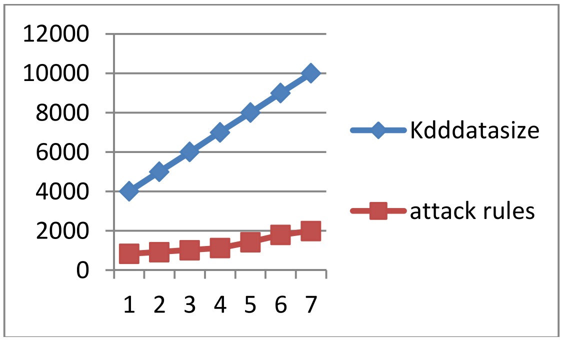


Figure 5: Represents The Integrated Data Size With Detected Attack Patterns Or Rules

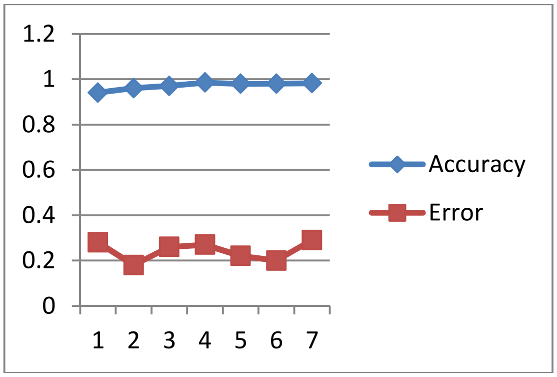


Figure 6: Represents The Integrated Data Size With Accuracy And Error Measures.

Table 3: Comparison of Accuracy of different models.

Threshold	Naïve Tree	ProbCorr Tree	Proposed Model
0.5	0.89	0.92	0.967
0.6	0.91	0.936	0.973
0.65	0.88	0.918	0.984
0.69	0.925	0.948	0.978
0.75	0.901	0.957	0.989

Table 3, describes the comparison between the proposed model with the traditional models such as Naïve tree and Probabilistic correlation tree in terms of time and accuracy.

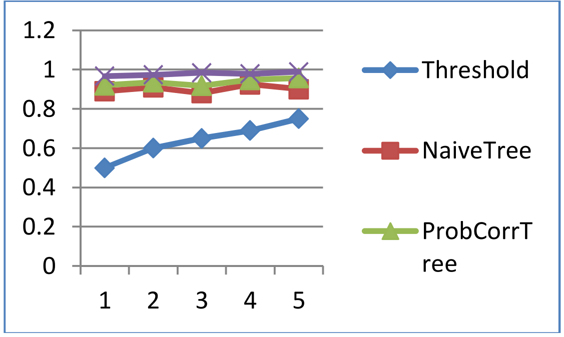


Figure 7: Comparison Of Proposed Model With Other Models

5. CONCLUSION

In this proposed work an improved dynamic pattern mining algorithm was implemented in the real time distributed data. This approach generates high quality attack patterns from the complex data. This approach minimizes the detection time and optimizes the decision patterns in distributed connected networks. This system generates intrusion patterns by integrating the predefined attacks and new attacks as early as possible with low false rate. Experimental results show that proposed dynamic model optimizes the real time true positive patterns with high accuracy compared to traditional models. In future, real time alert system using decision patterns will be evaluated to predict the attacks in the dynamic networks within the specified time interval.

REFRENCES:

[1] Weller-Fahy, D.J.; Borghetti, B.J.; Sodemann, A.A."A Survey of Distance and Similarity Measures Used Within Network Intrusion Anomaly Detection",Communications Surveys & Tutorials, IEEE,Year: 2015, Volume: 17, Issue: 1.

[2] Pontarelli, S.; Bianchi, G.; Teofili,"Traffic-Aware Design of a High-Speed FPGA Network Intrusion Detection System",Computers, IEEE Transactions on Year: 2013, Volume: 62, Issue: 11.

[3] Muradore, R.; Quaglia, D,"Energy-Efficient Intrusion Detection and Mitigation for Networked Control Systems Security", Industrial Informatics, IEEE Transactions on Year: 2015, Volume: 11, Issue: 3

[4] Bando, M.; Artan, N.S.; Chao, H.J.,"Scalable Lookahead Regular Expression Detection System for Deep Packet Inspection", Networking, IEEE/ACM Transactions on Year: 2012, Volume: 20, Issue: 3.

[5]Chunjie Zhou; Shuang Huang; Naixue Xiong; Shuang-Hua Yang; Huiyun Li; Yuanqing Qin; Xuan Li,"Design and Analysis of Multimodel-Based Anomaly Intrusion Detection Systems in Industrial Process Automation",Systems, Man, and Cybernetics: Systems, IEEE Transactions on Year: 2015, Volume: 45, Issue: 10

[6] Chittur, A, "Model generation for an intrusion detection system using genetic methods", Thesis, In coorperation with Columbia university,2001.

[7] Cohen, w. " Fast effective rule induction ", in proceedings 12th international conference on machine learning", pages 115, Fundamentals of database systems.

[8] B.A. Nahla, Salem, Zided,"Naïve bayes vs decision trees in intrusion detection systems", proceeding of the ACM symposium on Applied computing,2004.

[9] S.Chavan,K.Shah,N.Dave," Adaptive neuro-fuzzy instrusion detection systems", the proceedins of the international conference on IT:Coding and computing,pp.70-74.IEEE Computer society ,2004.

[10] B.Amor, S.Benferhat, "Naïve bayes vs Decision trees in intrusion detection systems," ACM symposium on applied computing, 2004, pp 420-424.