An adaptive and Scalable Scheme for Intrusion Detection

Anurag Jain [#1], Bhupendra Verma [#2], and J. L. Rana [#3]
1. *anurag.akjain@gmail.com*
2. bk_verma3@rediffmail.com
3. jl_rana@yahoo.com
Rajiv Gandhi Technical University, Bhopal

Abstract: Any anomalous activity could be indicative of intrusion. Previous researchers have been developed several techniques and algorithms based on this (anomaly detection) approach to detect intrusion. Now, anomaly based intrusion detection systems are widely used for intrusion detection. True positive and false positive parameters have been used to compare performance of all algorithms. However, depending upon the application a wrong true positive or wrong false positive may have severe detrimental effects. This necessitates inclusion of cost sensitive parameters in the performance.

In this paper, we motivate the need for cost sensitive analysis, and propose an approach for feature elimination and feature selection to reduce the cost involving processing of huge size of data such as KDD-CUP-99. Also, a method is proposed to select the elements of cost sensitivity metrics for further improving the results to achieve the overall better performance. The impact on performance trade off due to incorporating the cost sensitivity is discussed.

Further, we propose an idea of adoptive and scalable scheme for intrusion detection system (ASID) methodology to build a predictive model by integrating multiple models. ASID refers to a group of individual classifiers that are cooperatively trained on data set in a supervised classification problem. A number of popular algorithms have been investigated. Each of the algorithms outperforms all others in some select attack class. However, the ensemble (comprising of one or more of those algorithms) obtained through ASID, outperforms each of its constituent algorithms across all attack classes. Through simulations over KDD-CUP-99 dataset, we have shown that ASID along with the proposed feature selection method significantly improves the performance.

Keywords: Intrusion detection system (IDS), True positive (TP), False Positive (FP),

INTRODUCTION

Out of individually best intrusion detection algorithm for a particular class of attack, the main problem remains as how to go for tradeoff between the accuracy of detection and the time taken to detect with the constraint of available system hardware and the varying load conditions. After selection of set of best algorithm for each class of attach a unifying frame work is proposed and also an algorithm is developed which get the system load condition and then decides whether single or set of algorithms are to be launched for getting the optimal detection accuracy under different operating conditions [1,2]. For this purpose an Adaptive and Scalable Scheme for Intrusion Detection (ASID) has been develop which takes in to account the system computation load. Exhaustive simulation studies have been carried out and the results compared with the previous best results [3].

I. Data Preprocess for IDS

The best way to classify is the main objective of our work and it is tested by determining and analyzes to get high accuracy in the classification of attacks and training time in the KDD99 data set. It will also be attempted to learn a better way to classify each type of four attacks (Probe, Dos, U2R, R2L). Several researchers have used various concepts to reduce the features. The very obvious and basic concept that can be gainfully used could be the amount of information actually contained in the different features of KDD CUP 99 data. In our work maximum information gain ratio (entropy) calculation is made the basis for minimizing the number of features [4].
We calculate the entropy set with k different values given by:

$$\text{Entropy (Set)} = I\,(\text{Set}) = \sum_{i=1}^{k} P(value\ i).\log_2 P(value\ i)$$

In above formula probability of getting the i^{th} value representing by P(value i).
First we consider all the features and there after gradually reduce the number of features and compare the information gain. It is found that the change in information gain with all the features and with 18 to 20 features is almost same others are changed. In fact, only 20 features are above the average. This shows that the original database has data concentration in a small group of values [5]. Features that have result within a small group of values are little of significant to describe an attack behavior. This indicates that the original dataset may contain irrelevant data for the IDS and so it needs to be optimized. The ten most common attributes that different methods simultaneously selected to form the key set of attributes are shown in Table 1 for detection of different categories of attacks [6].

Table 1. Selected Attributes for Individual Attack Category

Attacks Selected	Attributes (as per sr. no of table 4.1)
DoS	3,4,5,6,8,10,13,23,24,37
Probe	3,4,5,6,29,30,32,35,39,40
R2L	1,3,5,6,12,22,23,31,32,33
U2R	1,2,3,5,10,13,14,32,33,36

These attributes form the reduced dataset on which the following intrusion detection methods are used.

II. Classifier Selection

Researchers have proposed several algorithms to detect an intrusion. Each of these algorithms has some advantage and disadvantage over the others and works satisfactorily for a limited types of intrusion. The need of the intrusion is design of an intrusion detection system which achieves high accuracy while lowering false alarm rate. The basic idea is to pickup set of the best algorithm covering all known intrusion types. In this section seventeenth such algorithms based on accuracy false alarm rate and computational time have been chosen [5].
In all seventeen algorithms were simulated using 10% of KDDCUP99 trend and test data set. Parameterized values of the results of various algorithms under comparison are present the graphical representation of the simulation result for parameters TP, FP and Accuracy rate of the classification of attacks in fig1, fig2, fig3.

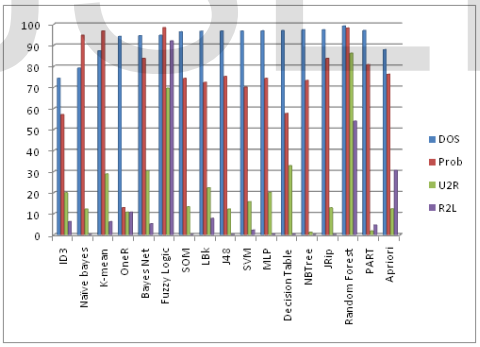

Fig 1. True Positive Rate in Different Classifiers

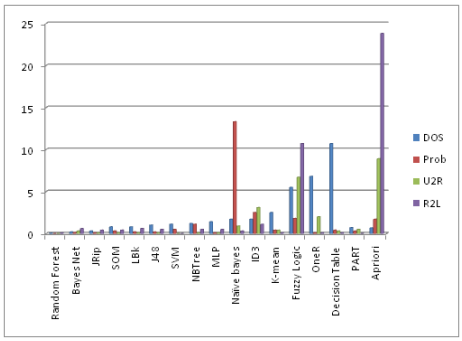

Fig 2. False Positive Rate in Different Classifiers

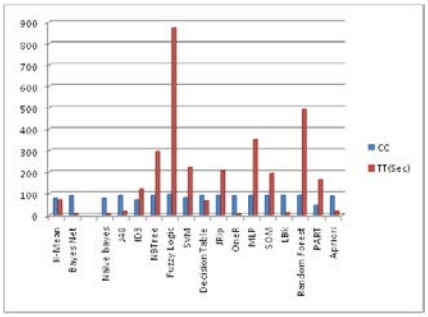

Fig 3. Average Detection and Total Time Taken

III. MODEL-1 AND MODEL-2

For model-1 selection of the algorithms is done based on the higher performance results (best True positive and worst false positive). It is clear that the following methods give the best results for each of the above 4 attack categories [5].

Model-2 based on minimum time taken that would take into consideration the need to quickly detect the intrusion and also the accuracy of detection with tradeoff in between depending on the prevailing needs. Therefore, we suggest a model with real-time algorithm selection. From the observation of the above model-2 has been proposed that primarily focuses on the detection of the intrusion in real time.

One thing is readily observable from the above he simulation results it that the proposed model -1 yields minor improvement in best TP compared to other single classifiers for DoS and Probe. Moreover significant improvement is obtained for detection of U2R and R2L attack categories while FP is reasonably small for all attack categories. Resource wise it is Model-2 which consumes minimal resources but the intrusion detection performance parameters are just satisfactory.

In numeric comparison of performance parameters Model-1 is superior with other approaches, however practically deployed in real systems there may have questions on some potential problems as following [7]:

- With hardcoded multiple algorithms deployment of a system is inflexible. And there is a remote possibility of best classifier being different if there the some data set which is better dataset than KDD99, the one used in this thesis, In such situation changing the classifiers may be much annoying.
- Resource requirement may be another problem when the models are implemented in very heavy workload/traffic situation. For example in multi-gigabit networks, running multiple algorithms for intrusion detection may hurt the entire system's network performance.
- And finally, how to make an optical choice of a set of algorithms in a multiple classifiers selection (MCS) system may be made when the proposed model is implemented. Especially when the load scenario is changing rapidly.

IV. MODEL-3 BASED ENSEMBLE CLASSIFIER

IDS provide some type of indication about an intrusion. Such indication (Alarm) may be true or false. For a particular algorithm there are different values for true and false alarm rate for different type of attacks. If the type of attacks is known before hand then the best IDS algorithm can be used to get a single decision. Always it may not be possible to use multiple algorithms. Consider a situation when we can run only a single algorithm because the computational resources eg. the CPU utilization is high (more then 75%). In such situation, suo moto it will be a single decision.

In case the best algorithm is very computation intensive then other remaining algorithm(s) of our proposed model (model-3) requiring much less computational power can be used selectively based on degree of computational load on the system. In such situation in may be necessary to go for sequential decision.

In the situation it is not known that what type of attack is to be expected and the system availability is high (CPU utilization is less than 25%) all the above four to five algorithms of model-3, can be used (full deployment) in the form of an ensemble. Formation of such ensemble becomes important because each of these methods has been trained using different training data set taken from different parts of feature space and also because attack may belong to any one of the heterogeneous set of features. The complete framework is shown in fig4.8.

Since different algorithms (each one best for a type of attack) are used in model-3, we shall use hybrid decision to combine the results of different algorithm voting with individual weigtage decided as per past history of the attacks.

- **About Model 3**

From theoretical view point it was necessary that what could be the best positive results under all conditions. For this we selected ensemble of seventeen algorithms and ran the simulation tests. The results are shown in table. xxx. It can be seen that this gives better results under any type of intrusion. The only drawback of this model is that it consumes by far the maximum resources. As such it is only for the theoretical interest not for practical implementation. Later we shall finally propose ASID framework that time simulation results shall be compared with the individual algorithm giving the max positive results and also with the results of model-3 algorithm.

- **General Consideration**

It is easily understandable that in actual implementation there has to be a system which would be not only accurate but also adaptable to the available resources. For this there next we propose a new "Adaptive and Scalable Intrusion Detection (ASID)" model using suitably modified/enhanced exiting algorithms to make entire intrusion detection system adaptive and scalable. The methodology is as following:

First a set of well known mining algorithms for pattern detection and pattern matching have been run to check the efficiency of individual algorithms in detecting a particular type of attack. The grading of these algorithms in detection was done from accuracy and resource requirement point of views. The best algorithm in each type of known intrusion detection was chosen and suitably modified with two objectives:

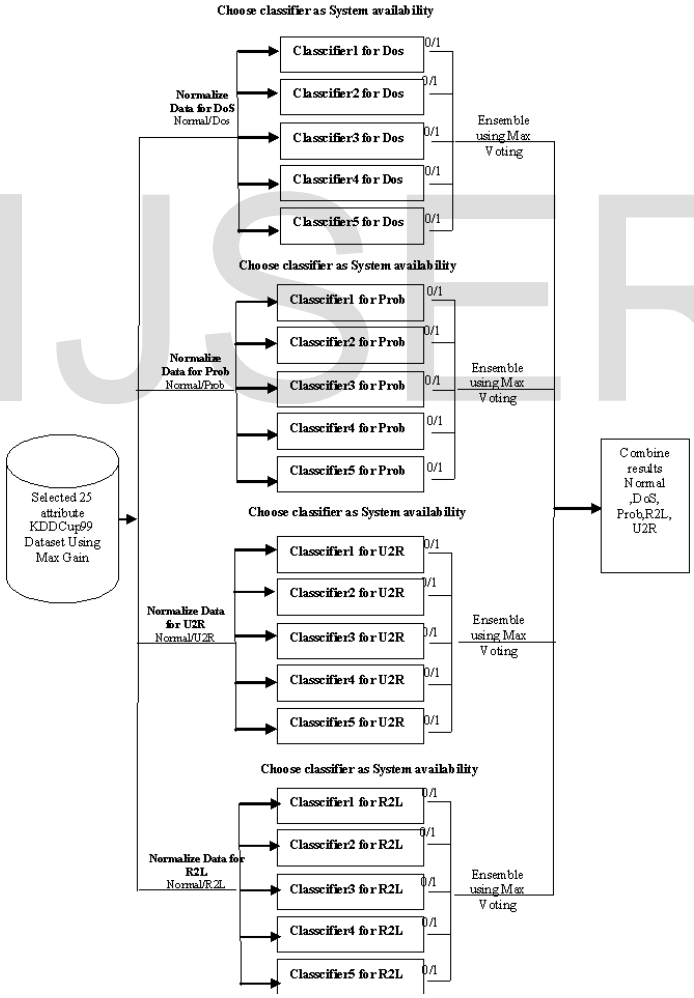

Fig 4.Adoptive and Scalable Intrusion Detection Model

a: multiple pattern matching using ensemble approach and combining the results by hybrid decision making approach. The weightage for individual algorithms was chosen based on the actual occurrence of different types of

intrusions during a predefined time window in our case past 1 hour. And for this all the used algorithms have been suitably modified and additional scripts were written to make the entire implementation run on WEKA.
b: The approach is adaptive i.e. to be able to adapt as per the system load and use scenarios i.e., less or more number of detectors (algorithms) can be launched so as not to burden the system owing to running of multiple algorithms such as model-3.
The ASID model architecture is such that optimal number of patterns/detectors can be easily deployed without causing much computational overhead which makes this approach scalable. The ASID architectural frame adaptively updates, deploys and combines multiple detection models as needed in the actual situation. Some intelligence in updating the weightage for results combination in ensemble is derived from record of intrusion pattern in an adjustable time windows.
KDD99 is the only dataset which is well tailored for simulation data mining. If the designed system is flexibly (no such thing as "hardcode the models"), then modifying the classifier selection models becomes easy. For the multiprocessor system individual algorithms of ASID can be configured to run in parallel to achieve real-time performance.

- **Simulation studies: Simulation of Model-3 For Various Algorithms**

Table 2 shows the selection of classifiers as best performance in terms of best TP and minimum FP for all class of attack.

Attack Type	Algorithm	TP	FP
DoS	a. Random Forest	99.2	0.05
	b. JRip	97.4	0.3
	c. MLP	96.9	1.4
	d. J48	96.8	1
	e. SVM	96.8	1.11
Probe	a. Fuzzy Logic	98.4	1.8
	b. Random Forest	98.2	0.01
	c. K-mean	96.8	0.04
	d. Bayes Net	83.8	0.13
	e. JRip	83.8	0.1
R2L	a. Fuzzy Logic	92.1	10.7
	b. Random Forest	54	0.09
	c. OneR	10.7	0.1
U2R	a. Random Forest	86.2	0.017
	b. Fuzzy Logic	69.6	6.7
	c. Decision Table	32.8	0.3

For generation of final result algebraic combiners were studied. It was discovered that the weighted majority voting algebraic combiner gives the better results. The occurrence of different types of attacks within the time window of immediate past 1 hour was used in deciding the weights.

V. Comparison of results of all the three models:

The graphical representation of results of all the models model- 1, model-2 and the finally proposed model-3 are summarized in fig4 and fig5.

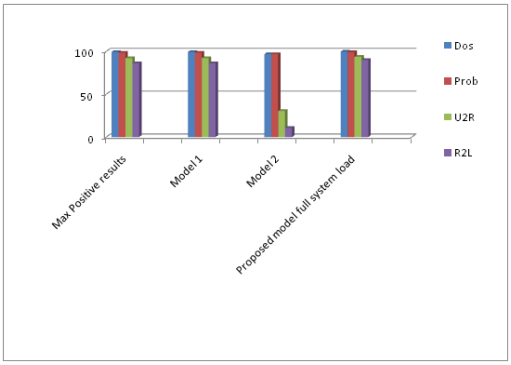

Fig4: TP for max positive and model1, Model2 and Model3

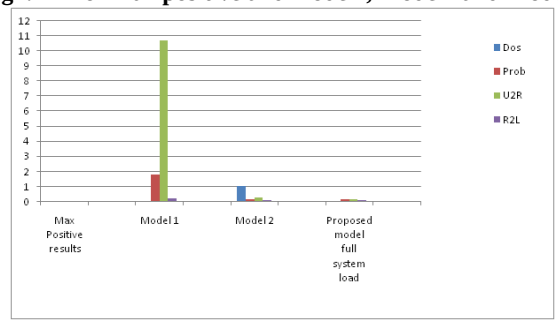

Fig5: TP for max positive and model1, Model2 and Model3

Conceptual comparison of model 1, model 2 and model 3:

Table 3 shows the conceptual comparison of model1, model2, and model3

Model 1	Model 2	Model 3
Only those classifiers are included which give best results of a particular class of attack.	Only those classifiers are included which for a class of attach take minim time and give good results.	All classifiers which give better than 75% performance in a particular class of attacks.
Tales intermediate computation time.	Takes minimum computation time.	Takes highest commutation time in all three models.
Non adaptive but scalable.	Non adaptive but scalable.	Adaptive and scalable.

- **General Consideration**

It is easily understandable that in actual implementation there has to be a system which would be not only accurate but also adaptable to the available resources. For this there next we propose a new Adaptive and Scalable Intrusion Detection (ASID) model using suitably modified/enhanced exiting algorithms to make entire intrusion detection system adaptive and scalable. The methodology is as following:

First a set of well known mining algorithms for pattern detection and pattern matching have been run to check the efficiency of individual algorithms in detecting a particular type of attack. The grading of these algorithms in detection was done from accuracy and resource requirement point of views. The best algorithm in each type of known intrusion detection was chosen and suitably modified with two objectives:

a: multiple pattern matching using ensemble approach and combining the results by hybrid decision making approach. The weightage for individual algorithms was chosen based on the actual occurrence of different types of intrusions during a predefined time window in our case past 1 hour. And for this all the used algorithms have been suitably modified and additional scripts were written to make the entire implementation run on WEKA.

b: The approach is adaptive i.e. to be able to adapt as per the system load and use scenarios i.e., less or more number of detectors (algorithms) can be launched so as not to burden the system owing to running of multiple algorithms such as model-3.

The ASID model architecture is such that optimal number of patterns/detectors can be easily deployed without causing much computational overhead which makes this approach scalable. The ASID architectural frame adaptively updates, deploys and combines multiple detection models as needed in the actual situation. Some

intelligence in updating the weightage for results combination in ensemble is derived from record of intrusion pattern in a adjustable time windows.

KDD99 is the only dataset which is well tailored for simulation data mining. If the designed system is flexibly (no such thing as "hardcode the models"), then modifying the classifier selection models becomes easy. For the multiprocessor system individual algorithms of ASID can be configured to run in parallel to achieve real-time performance.

VI. An adaptive and scalable and scalable scheme for Intrusion Detection System (ASID) Concept:

For use with ensemble approach in model 3 a novel technique for CPU load balancing in real time is proposed. The proposed adaptive algorithm is a combination of existing dynamic priority driven algorithm i.e. adaptive and scalable scheme for intrusion detection system (ASID). AISD Algorithm for choosing scheduling algorithm has been tested for both under and overloaded CPU conditions. During testing initially, the process was started with assumption of an under loaded condition. When CPU load is gradually increased the ASID scheduling during overloaded condition changed the choice of algorithms as per CPU Utilization. Thus the proposed algorithm is adaptive as it uses the strong features of ensemble of algorithms based on the computation load on the system thus overcoming the drawbacks of CPU getting overloaded. We have simulated, proposed adaptive algorithm along with algorithms for real time systems. %Success Rate and %Effective CPU Utilization are used as performance measuring criteria for AISD. The evaluation of results and comparison of our proposed adaptive CPU scheduling algorithm with CPU utilization algorithm shows that the proposed adaptive algorithm is optimal and efficient during under loaded as well as overloaded situations compared to AISD.

- **Consideration of Processor Utilization**

Many tools are available to determine a least upper bound to processor utilization in multitasking system with fixed priority [8]. We define the (processor) utilization factor to be the sum of the fraction of processor time spent in the execution of each task of the task set. In other words, the utilization factor is equal to one minus the fraction of idle processor time. Since C_i/T_i is the fraction of processor time spent in executing task ζ for m tasks, the utilization factor is:

$$U=\sum_{i=1}^{m} Ci/Ti$$

we shall use $\zeta_1\ \zeta_2\ \zeta_3$ …… ζ_m to denote m periodic tasks, with their request periods being T_1, T_2, ... , T_m and their run-times being C_1, C_2, ... , C_m respectively. The request rate of a task is defined to be the reciprocal of its request period. The normal upper bound on CPU utilization is around 80% and the low CPU utilization around 25%. At any given time CPU utilization can be examined by using following standard JAVA Program embedded in the WEKA script [95] as given below.

- **Classifier Combination**

Individual classifiers work best for select type of intrusion as can be seen. The basic idea behind using multi classifier is to take benefit of the detection power of individual classifiers. The next problem is how to combine the results of individual classifiers to produce the overall better results.

The option that has been chosen is governed by following factors:

1. It should be possible to take advantage of knowledge generated previously to guide the combining in the next iteration.
2. The individual classifiers are performing the same task and some classifiers almost have same performance for a type of intrusion but for other type of intrusion other classifiers are best suited. This consideration calls for techniques of combiners that use much less memory and computation time.

- **Performance based Priority assignment:**

For all four attack types and for each of the classifier algorithm in that attack category, performance criteria PC_1, PC_1= (TP-FP)/100 is calculated and the priorities are assigned as per higher values of PC_1. In case two algorithms yield the same PC_1 value , another performance criterion PC_2, PC_2=TT/Max(P2) is calculated and out of algorithms having same PC_1 values the algorithm which has lower PC_2 value , higher priority is assigned.

- **AISD Algorithm: Functioning**

Within the model 1 to model 3, the AISD scheduling algorithm is a priority driven algorithm. The one or more classifiers ensembles are used as per the priority and the computation load on the system. For intrusion detection, before a new classifier is initiated from the ensemble, the then processor utilization status is checked. If sufficient computation scope is available one after another various classifiers of a model are in-acted and the final result of the ensemble is generated. In case when sufficient processor time is not available AISD launches the lower priority classifiers. In heavily loaded system only classifiers of model 2 may be launched.

Each ensemble model has set of algorithms and it works as per following:

• During under loaded condition, the AISD algorithm uses all or some of the classifiers based on priority and the extent of CPU load decided dynamically.

• During overloaded condition again based on the extent of CPU load following actions are taken:

1. For intermediate system load Model 1
2. For heavily loaded system model 2 is launched or
3. For lightly loaded system one or more than one classifier from each of the attack type again based on priority can be used.

In AISD model we divide two algorithms first is simple AISD and other is complex AISD.

- **Algorithm for Simple ASID**

step1: if CPU is max overloaded (>80%) then we use Model-2 (Based on Min TT)

Step2: if CPU is intermediate or partially overloaded (>40 %< 80%) then we use model-1 (Based on Max TP and Min FP)

Step3: if CPU is Min Overloaded (<40%) then we use full or partial model-3 (Complex AISD**)** based on ensemble of classifier) // This is Algorithm of Complex AISD which is explained below:

- **Algorithm of Complex ASID**

First we select five algorithm as per above calculated maximum PC1 or only those algorithm which have PC1>0.5 // Next Calculation of PCC:

Step1: create set of all n PC2 in all class of algorithms where n is number of algorithm in a particular attack class i.e. ($PC2_1$, $PC2_2$, $PC2_3$.....$PC2_n$).

Step2: find the power subset 2^n of given n sets (a set of PC2, and a collection of n elements, each with a positive integer weight $PC2_i$)

{$PC2_1$},{$PC2_2$},{$PC2_3$},.....{PC2n)},{($PC2_1$,$PC2_2$)},.{($PC2_1$,$PC2_2$,$PC2_3$.....$PC2_i$)}.....{($PC2_1$,$PC2_2$,$PC2_3$.....$PC2_n$)},.

Step3: find PCC of all power subsets :

Find a power set S of given set such that

PCC_i= $\sum_{i \in S} w_i$ Where i represents the ith subset of given set S.

For example sum all subsets of Step2 (total 2^n Number of PCC),

if single element subset then

PCC_1=$PC2_1$,PCC_2=$PC2_2$,PCC_3=$PC2_3$ .....PCC_n=$PC2_n$

if sub set has two elements then

PCC_{12}= $PC2_1$+$PC2_2$,PCC_{13}=$PC21$+$PC2_3$..... $PCC_{(n-1)n}$=$PC2_{n-1}$+$PC2_n$

if subset has three elements then

PCC_{123}=$PC2_1$+$PC2_2$+$PC2_3$,PCC_{124}=$PC2_1$+$PC2_2$+$PC2_4$.....

$PCC_{(n-2)(n-1)n}$=$PC2_{(n-2)}$+$PC2_{(n-1)}$+$PC2_n$

likewise, if subset has n elements then

$PCC_{12.....n}$= $PC2_1$+$PC2_2$+$PC2_3$.....+$PC2_n$

Step4: sort all PCC in increasing order.

Step5: calculate CPU utilization as per incremented scaling K part (in our case K=10) i.e. if CPU utilization is calculated in increment of 10% as such there are 10 points to check. Let CPU utilization in %ge be represented by CU in increment of 10%.

Step6: if CU= 1,2,3.....K then // for K check points

a. select PCC (performance classifier as in step4)

b: if number of PCC ≥ K then

$K=2^n$ use all combination of PCC in incremented CU e.g. in CU_i we use Combination of PCC_i

else

$R= 2^n$ // R is total no of PCC

then

G=INTEGER [K/R] // calculation of Gap G

Then we use $PCC_{r \times G}$ Where r = 1,2,3……..K

end if

step7: Use step 1to 6 in all class of attack.

step8: Combine result and generate alarm if attack.

Step9. Exit // sample hand calculation for n=5 and K=2 in appendix A

VII. Result Discussion

Discussion of the results for Simple ASID

It can be observed that in all cases and all types of attacks the results of model-3 match and surpass the best result. This has been possible because of use of ensemble approach. This simply means that even if the type of attack is not

known apriori, from mixture of different types of attacks model-3 can detect any attack with the same or better accuracy as that of the best detection efficiency of a specific algorithm (for which prior knowledge of type of attack is necessary) of a particular type of attack .

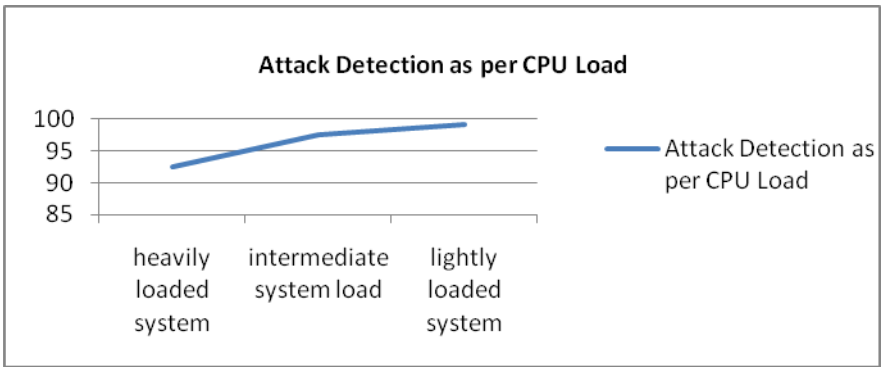

Fig 6 Performance Based Priority Assignment Considering Available Computation Power

Discussion of the results for Complex ASID

In complex ASID, first, priorities PC_1 (representing accuracy) and PC_2 (representing response-time) are computed for each of the 17 classifiers. Next, for each type of attack (DOS, PROB, R2L and U2R), up to 5classifiers are selected if max PC_1 value is greater than 0.5. Thus, following numbers of classifiers have been chosen.

ASID's adaptive characteristics allows for enhancement of the above chosen classifier list by allowing evaluation of more classifiers, resulting in larger number of chosen-classifier list and/or more stringent criteria of selection giving even better performing classifiers in the chosen list.

CPU availability is divided into 10 parts on a scale of 0 to 100 (i.e., part 1 corresponding to lowest CPU availability and 10 corresponding to highest CPU availability). For each attack type, from its chosen list of classifiers, all possibilities- 31 combinations for DOS, 31 for PROB, 3 for R2L and 3 for U2R are considered and performance computed. These chosen performance values are used by ASID to select classifiers as the CPU availability scales.

When CPU availability is maximum (i.e., CU = 10), all classifiers from the chosen list are applied. However, as the CPU get busier and CU values get lower, classifiers starting from highest PC_2 value are dropped as per complex ASID algorithm.

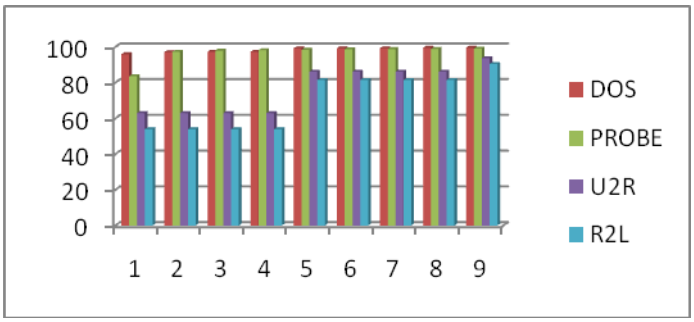

Fig7 compression TP rate for Available Computation Utilization (CU)

The above fig7 gives TP values for various attacks detected by list of classifiers selected by the ASID algorithm. where, X axis gives CPU availability (CU) and Y axis represents TP rate. It can be seen that compared to single classifier, ASID algorithm gives significant improvement in TP values. In fact, when more CPU resources are available the performance improves very significantly.

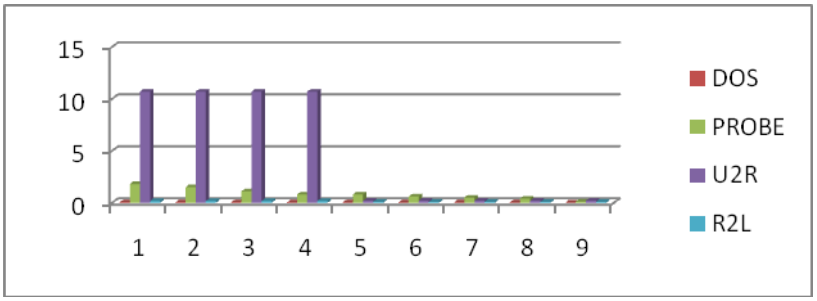

Fig8.Compression FP Rate for Available Computation Utilization (CU)

In the fig8 , FP values for various attacks detected with the help of ASID algorithm have been given. Compared to single classifier. The classifiers chosen and dynamically managed by ASID algorithm have much lower (thus better) values of FP. A pictorial depiction of the same (FP V/S CU) has been given in figure 9.

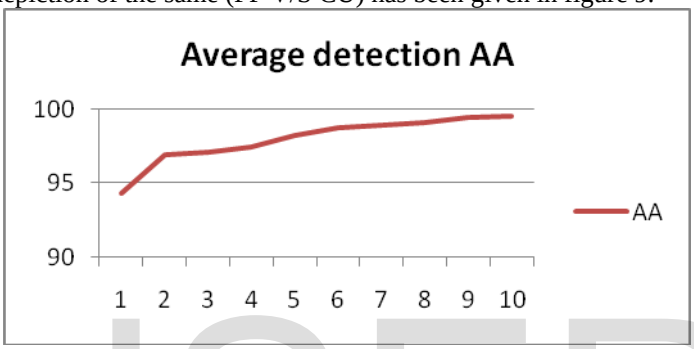

Fig 9. Average attack detection as per CPU load

In the figure 9, it can be seen that the AA (Average Accuracy) values improve, as expected, with increase in the availability of CPU resources (CU). The AA values from ASID algorithm are significantly better compared to the AA values for single classifier.

VII. CONCLUSION

As the volume of e-commerce is increasing and the safely of other data have become paramount, the need for intrusion detection has increased by many folds in last few years. Now commercially more money is being invested for development of algorithm to detect intrusion as accurately as possible. Additionally there is need for detection at the minimum time. The hardware has become more powerful and when the question of security comes other costs become secondary.

In this work various methods, framework and approaches used by researcher have been systematically grouped and strength and weakness have been identified. The proposed ASID gives the optimal detection for a given server conditions. The results have better or same accuracy as of the overall all best reported algorithms for detecting that attack type. Normally such algorithms are computationally demanding and may cause CPU overload problem. The purposed ASID model takes care of such problems by judging in real time the load on CPU and accordingly launching computationally less heavy algorithms (marginally less accurate in detection).

At the end, a methodology have been very briefly discussed that would assimilate various aspects of intrusion detection. Normally clubbing multiple approaches consumes more resources thereby slowing down the system especially in real time intrusion detection. To avoid these in the ASID technique for load balancing has been incorporated so that the intrusion detection tool does the self-adjustment and overloading of the system under scan is avoided.

Several researchers have proposed good algorithms for intrusion detection. In this thesis all the best performing algorithms have been chosen accuracy wise and computation time wise. A frame work has been proposed which has five most accurate detecting algorithms and five least computation time consuming algorithms.

These algorithms are launched for intrusion detection depending upon the current load on the system. Along with this frame work a selection algorithm (ASID) has been proposed. Depending on the current system load ASID decides whether a single algorithm or combination of set of algorithm should be launched so that without causing any overload to the system the optimal detection is achieved both accuracy wise and also computational time wise. The best part of it is that with the system capabilities the detection accuracy will also improve as now better/bigger ensemble of algorithms can be launched.

For normalizing the KDDCUP 99 dataset, entropy based information gain ratio calculation for different attributes were individually carried out for all 41 features. Class wise gain ratios were analyzed and set of 10 attributes for each class of attacks was chosen to reduce the computation time.

There is scope for fine tuning the proposed frame work and ASID. In the situation where load on the system is highly dynamic, proposed framework and ASID should probe for the system load frequently and modify the launching of algorithms accordingly. Moreover level of detection accuracy and the detection time can also be pre specified for ASID to decide the proper launching of detection algorithms to meet the set targets. Some more work needs to be done to intelligently adjust the weight in machine learning as well as in modifying the implemented result of individual technique in ensemble approach when final inferences are drawn. Whatever tools are used in intrusion detection, results shall not be 100% correct always. Therefore it is also necessary to incorporate cost factor in intrusion detection to penalize the wrong result and/or reward the correct ones depending on the nature of work server is handling.

REFERENCES

[1]. Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. "Intrusion Detection System: A Comprehensive Review." Journal of Network and Computer Applications 36, No. 1 (2013), PP16-24.

[2]. Monali Shetty and N. Shekokar "Data Mining Techniques for Real Time Intrusion Detection Systems", International Journal of Scientific & Engineering Research Volume 3, Issue 4, April (2012), PP 764-770.

[3]. A. Jain, B. Verma, & J. L. Rana, “Anomaly Intrusion Detection Techniques: A Brief Review”. International Journal of Scientific & Engineering Research, 5(7), pp. 1372-1383.

[4]. Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu and Ali-A. Ghorbani "A Detailed Analysis of the KDDCup 99 Data Set", In Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications 2009.

[5]. Anurag Jain, Bhupendra Verma, and J. L. Rana. "CLASSIFIER SELECTION MODELS FOR INTRUSION DETECTION SYSTEM (IDS)" Informatics Engineering, an International Journal (IEIJ), Vol.4, No.1, March (2016): 1-11., 2014.

[6]. Wei Wang, Sylvain Gombault and Thomas Guyet "Towards Fast Detecting Intrusions: Using Key Attributes of Network Traffic", The Third International Conference on Internet Monitoring and Protection, IEEE 2008, ICIMP08, PP 86-91.

[7]. Özge Cepheli, Saliha Büyükçorak, and Güneş Karabulut Kurt, “Hybrid Intrusion Detection System for DDoS Attacks” Journal of Electrical and Computer Engineering Volume 2016 (2016), Article ID 1075648, 8 pages.

[8]. Martin Schoeberl "A Java Processor Architecture for Embedded Real-Time Systems", Journal of Systems Architecture 54, No. 1, 2008, PP 265-286.