Australian Journal of Basic and Applied Sciences, 3(3): 2581-2597, 2009
ISSN 1991-8178

Anomaly Intrusion Detection System Using Information Theory, K-NN and KMC Algorithms

Hossein M. Shirazi

Malek-Ashtar University of Technology, Faculty of ICT, Tehran, Iran

Abstract: The huge expansion in the size of internet and the appearance of various types of malicious attack and the insufficiency of traditional security tools like antivirus conduct to the development of intrusion detection systems or IDS. Since current IDSs are signature based they stil are unable to detect new forms of attacks even if these attacks are slightly derived from known ones. So, recent researches concentrate on developing new techniques, algorithms and IDSs that use inteligent methods. In this paper some anomaly detection engines have been proposed based on K-NN, K-Means Clustering (KMC) algorithms. Firstly, by applying information theory measures, network connection features were ranked according to their importance in detecting attack classes like DOS, R2L, U2R, and PROBE. This ranking strongly helped in selecting only the most important features and by consequent decreasing dramaticaly the computation complexity and eliminating the noise resultant of irrelevant features. Secondly, some anomaly Intrusion Detection models approaches were proposed depending on features selection, k-NN and k-means clustering. These approaches proved their efficiency, where the detection rate was more than 92% and was better than other approaches especialy in detecting dangerous attacks like R2L and U2R.

Key words: Intrusion Detection System, K-NN, K-MEANS Clustering, KDD99, Features Selection

INTRODUCTION

With the increasing creativity of attackers, the development of totaly effective security tools is impossible. Indeed Current security tools like antivirus are now not sufficient to perform the institute security integrity. The need to accurately detect malicious intrusions has arisen and intrusion detection is an increasingly important technology that monitors network traffic and recognize ilegal use, misuse of computer systems, and malicious attack to computer systems.

The difference between security products and intrusion detection systems is that the latter needs more inteligence. They must analyse al gathered information and deduce useful results.

IDSs are either signature based or anomaly based. Most of current IDSs are signature based, in which they search in the captured traffic or log files for a signature and send an alarm. This is similar to antivirus work methodology.

Anomaly based IDSs measure the deviation in behavior of normal system state, and then sends an alarm if the deviation exceeds a certain threshold.

Besides, IDS can be classified as Network IDS (NIDS) or host based IDS (HIDS). NIDS deals with network traffic while HIDS deals with computer system logs and cals.

A. Information Theory Concepts:

According to Athanasios and Pillai (2002), the heuristic interpretation of entropy, the number H(U) entropy of partition U is a measure of our uncertainty about the event Ai of the partition U prior to the performance of the underlying experiment. If the experiment is performed and the results concerning Ai become known, the uncertainty is removed. We can say that the experiment provides information about the event Ai equal to the entropy of their partition.

- Entropy measure: For a random variable X with n outcomes $\{x_i: i= 1,n\}$, the Shannon entropy, a measure of uncertainty and denoted by H(X) is defined as:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

Corresponding Author: Hossein Shirazi, Malek-Ashtar University of Technology, Faculty of ICT, Tehran, Iran
E-mail: Shirazi@mut.ac.ir

Where $p(x_i)$ is the probability mass function of outcome x_i.

- Conditional entropy: The conditional of X assuming Y is measured by:

$$H(X|Y) = -\sum_{i=1}^{Nx} p(x_i|Y)\log_2 p(x_i|Y)$$

Where $P(Y) \neq 0$, NX is the elements X_i of X and

$$P(x_i|Y) = \frac{P(x_i \cap Y)}{P(Y)}$$

$H(X|Y)$ is the uncertainty about U in the subsequence of trials in which M occurs.

- If Y is partitioned into NY partition then

$$H(X|Y) = -\sum_{i=1}^{Ny} p(y_i)\log_2 p(y_i)$$

$$H(X|Y) = -\sum_{i=1}^{Ny}\left\{ p(y_i)\sum_{i=1}^{Nx} p(x_i|y_j)\log_2 p(x_i|y_j)\right\}$$

$$H(X|y_i) = -\sum_{i=1}^{Nx} p(x_i|y_j)\log_2 p(x_i|y_j)$$

- Mutual information (important): is measured by:

I(X,Y) = H(X) - H(X|Y)
I(X,Y) = H(Y) - H(Y|X)

From the above relation we see that mutual information can be interpreted as "the amount of decreasing in uncertainty about X when Y is given"

Mutual information is "Information about X contained in Y" Or "Information about Y contained in X"

B.K-NN Algorithm:

In pattern recognition, the k-nearest neighbor's algorithm (K-NN) is a simple method for classifying objects based on closest training examples in the feature space.

The k-nearest neighbors algorithm is a simple machine learning algorithms. An object is classified by a majority vote of its neighbors. k is a positive integer, typicaly smal. If k =1, then the object is simply assigned to the class of its nearest neighbor.

The k-nearest neighbors are determined according to a distance function like the Euclidian distance. To determine neighbors of an object, we calculate the distances from this object to the whole points in a reference data. Then we sort these distances in an ascending order and select top k points, these are caled k-nearest neighbors.

C. K-means Clustering Algorithm:

The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k clusters. Each cluster has a center which is the mean value of the objects in a cluster, this mean value can be viewed as the cluster's centroïd or center of gravity. The use of clustering is that instead of dealing with al objects in clusters, we can just deal with cluster's centroïds. This reduces very much the complexity. Fig. 1 shows the KMC algorithm.

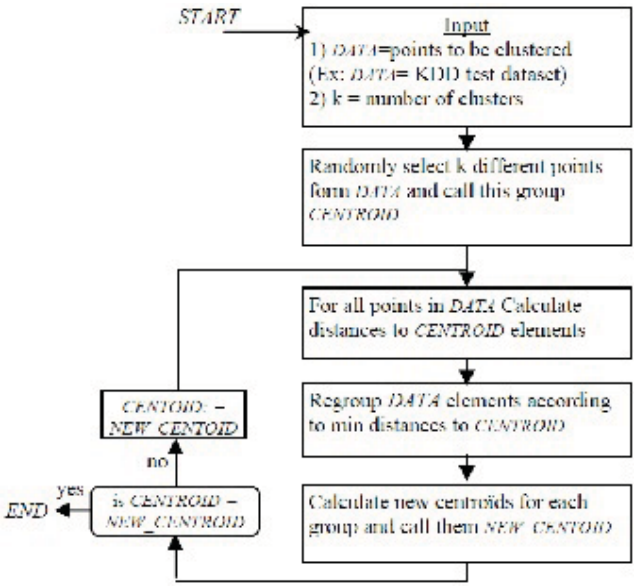

Fig. 1: K-means clustering algorithm

II. Background of Similar Work:

In Middlemiss and Dick paper (2002), the authors have implemented a simple genetic algorithm which evolves weights for the features of the data set. Then k-nearest neighbor classifier was used for the fitness function of the GA as wel as to evaluate the performance of the new weighted feature set. The main aim of work was to rank features according to there importance. The results provided an increase in intrusion detection accuracy.

Liwei Kuang (2007), proposed a Dependable Network Intrusion Detection System (DNIDS) based on the Combined

Strangeness and Isolation measure K-Nearest Neighbor (CSIKNN) algorithm. The intrusion detection algorithm analyzes different characteristics of network data by employing two measures: strangeness and isolation. But in general the K-NN stil needs intensive computations.

The Unsupervised Anomaly Detection Using an Optimized K-Nearest Neighbors Algorithm can work without the need for massive sets of pre-labeled training data. The author discussed the creation of such a system that uses a k-nearest neighbors algorithm to detect anomalies in network connections, as wel as the optimization necessary to make the algorithm feasible for a real-world system. The drawback of this approach is that the detection rates and false positive rates were not good as other approaches.

Xiao *et al*. (2005) present an approach that uses information theory and genetic algorithms to detect abnormal network behaviors. The information theory was used to determine the most relevant features to the detection operation. A smal number of network features are closely identified with network attacks. However, the approach considers only discrete features and ignored other features which are very important like duration, source and destination bytes.

A.M and A.S (2008) proposed a technique of combining K-Means clustering and genetic algorithm to IDS. The training data has been clustered into 2-clusters before feeding the initial population hoping that data wil be divided into normal and abnormal clusters. There were no declared experiment results but the author concluded that his approach detected known and unknown and the results were not good for some runs.

III.KDD99 Data Sets Description:

The KDD99 dataset is now the benchmark for training, testing and evaluating learning IDSs, so it is basic for IDS developers.

A. KDD99 Dataset Description:

The dataset was used for the Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD-99, the Fifth International Conference on Knowledge

Discovery and Data Mining. The competition task was to build a network intrusion detector, a predictive model or a classifier that can tel what are "bad" connections, caled intrusions or attacks, and what are "good", caled normal connections.

Lincoln Labs set up an environment to acquire nine weeks of raw TCP dump data for a local-area network (LAN) simulating a typical U.S. Air Force LAN. They operated the LAN as if it were a true Air Force environment, and inserted multiple attacks. These attacks fal into four main categories:

- DOS: denial-of-service, e.g. syn flood.
- R2L: unauthorized access from a remote machine, e.g. guessing password;
- U2R: unauthorized access to local super user (root) privileges, e.g., various "buffer overflow" attacks;
- Probing: surveilance and other probing, e.g., port scanning.

The KDD99 datasets is composed of records that are caled connections. Each connection has 41 features and a label which indicates the attack name. The label is supported only for the training dataset. Fig. 2 shows these features.

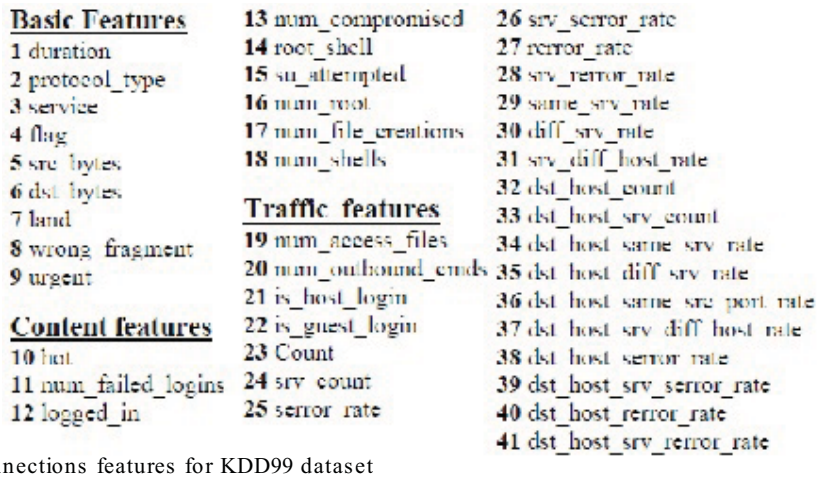

Fig. 2: The 41 connections features for KDD99 dataset

The KDD99 datasets are divided into three parts:

- Ful training dataset which has 4898431 labeled records and is used for training purposes.
- 10% training dataset which has 494021 labeled records. Since the ful dataset is huge an IDS developer can train his IDS on just 10% of the ful dataset.
- Testing dataset. It has 311029 unlabeled connections. Table I shows the distribution of these connections.

Table I: Distribution of Attack Classes

Class	10% KDD Training Dataset		Full Training Dataset		Testing Dataset	
normal	97278	19.69%	972781	19.86%	60593	19.48%
DoS	391458	79.24%	3883370	79.28%	229853	73.90%
Probe	4107	0.83%	41102	0.84%	4166	1.34%
R2L	1126	0.23%	1126	0.02%	16347	5.26%
U2R	52	0.01%	52	0.001%	70	0.02%

B. Attack Types Distribution:

In the folowing, Table II shows the known and unknown attacks in both training and testing KDD99 datasets.

Table II: Known and Unknown Attacks in Training and Testing Datasets

Class	Known Attacks (in training + testing data sets)	unknown Attacks (in testing data set)
DoS	back, land, neptune (syn flood), pod (ping of death), smurf, teardrop	apache2, mailbomb, processtable, udpstorm
R2L	ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster	httptunnel, name, sendmail, snmpguess, worm,xlock, xsnoop
U2R	buffer overflow, loadmodule, perl, rootkit	ps, sqlattack, xterm
PROBE	ipsweep, nmap, portsweep, satan	mscan, saint.

IV. Features Selection:

Indeed it is very important to reduce the number of features by selecting the most important ones. For example in the problem of designing an inteligent intrusion detection system, developers are exposed to deal with the KDD99 data sets, the training data set contains about 5 milion connection records, each record contains 41 features. So it is time consuming to train the system considering al features. But if we could reduce the number of features for example to 8 features we wil decrease very much the amount of huge computations.

Our approach wil employ the information theory to sort the most important features.

Indeed it is thought that data set space can be divided into:

- Broadly 2 main classes: Normal & Attack.
- Or more precisely 5 classes: Normal, DoS, R2L, U2R, Probe

So the dataset space can be described by the folowing random variables:

X: the decision random variable. Its state space is {Normal, DOS, U2R, R2L, and Probe}

Y: The connection feature random variable. Indeed there are 41 independent random variables like

- $Y = Y_{Protocol\_type} = \{ICMP, TCP, UDP\}$
- $Y = Y_{land} = \{0, 1\}$

Now to extract the importance of a connection feature (ex. ptotocol_type), one can calculate "the amount of information about X (normal connection or DOS or R2L or U2R or Probe) contained in Y (connection feature ex. protocol_type)"

Indeed the previous statement is exactly the mutual Information (Athanasios Papoulis and S. Unnikrishna Pillai, 2002) of X and Y that are explained in I-A.

A. Case Study: Information about Attacks Contained In Protocol_Type Feature:

The folowing, is a case study example to give an explanation about how to calculate mutual information for X {normal, Attack} and Y{UDP,TCP,ICMP}.

Step1: we calculate the distribution of connections of X and Y using the KDD99 training Ful dataset as shown in Table III.

Table III: Conn# According to Protocol type and Traffic Class

	X random variable					
	Normal	DoS	R2L	U2R	PROBE	Total
Yprotocol_type						
ICMP	12763	2808150	0	0	12632	2833545
TCP	768670	1074241	1126	49	26512	1870598
UDP	191348	979	0	3	1958	194288
TOTAL	972781	3883370	1126	52	41102	4898431

Then we calculate the probability distribution of these connections as shown in Table IV.

In Table IV, The upper probability value in italic is the intersection probability P(X, Y). The lower probability value is the conditional probability P (X|Y) = P(X,Y) / P(Y).

Table IV: Probability and Conditional Probability of X and Y=yprotocol_type

	X random variable					
	Pnormal	PDoS	PR2L	PU2R	PPROBE	P(Y)
Yprotocol_type r.v						
PICMP	0.0026	0.5733	0	0	0.0026	
	0.0045	0.9910	0	0	0.0045	0.5785
PTCP	0.1569	0.2193	0.0002	1.0003E-05	0.0054	
	0.4109	0.5743	0.0006	2.6195E-05	0.0142	0.3819
PUDP	0.0390	0.0002	0	6.1244E-07	0.0004	
	0.984	0.0050	0	1.5441E-05	0.0101	0.0396
P(X)	0.6140	0.3581	0.0008	3.6811E-05	0.0270	

Step2: Let's calculate the entropy H(X) of X which means the uncertainty of X (normal or attack)

$$H(X) = -P_{normal}\log_2 P_{normal} - P_{attack}\log_2 P_{attack}$$

where $P_{attack} = P_{DoS} + P_{R2l} + P_{U2R} + P_{PROBE} = -1 - P_{normal}$

H(X) = -0.614 log (0.614) - 0.3851 log (0.3851) = 0.9622

Step3: Let's calculate the conditional entropy H(X, connection feature) of X which means the uncertainty of X (normal or attack) given that the connection feature is protocol_type.
H(X|Yprotocol_type)=P(icmp){H(normal|icmp)+H(attack |icmp)}+p(tcp) {H(normal|tcp)+H (attack|tcp)}+P (udp) {H(nor mal |udp) +H(attack |udp)}
H (normal |icmp) =-p (normal |icmp) log P (normal |icmp)
H(X |protocol_type) = 0.5785 {-0.0045 log (0.0045) -0.9955
log (0.9955)} +0.3819 {-0.4109 log (0.41092)-0.5891 log
(0.5891)} +0.0397{-0.9849 log (0.9849) - 0.0151 log (0.0151)} = 0.4016
So the Uncertainty decreased from 0.9622 to 0.4016 when
protocol_type is introduced
Step4: calculating the mutual information of X and Y
I(X, Y) =H(X)– H(X|protocol_type) =0.9622-0.4016 = 0.5606

So 0.5606 is the amount of information about X {normal or attack} contained in protocol_type

In the same way one can get the I(X,Y=Yprotocol_type) for X={DoS, Others}, {R2L, Others}, {U2R, Others}, {PROBE, Others}.

B. All Features Ranking:

In case study we have studied the role of protocol_type feature in classifying a connection into one of the following classes {Normal, DoS, U2R, R2L, and PROBE}. Now we have to generalize previous calculation for the 41 features. So Table III, Table IV and mutual measures I(X,Yfi) will be recalculated 41 times. The resulting calculations will be organized in a table like Table V.

Table V: Mutual Information Table

Feature	Normal /attack	DoS/Others	R2L/Others	U2R/others	probe/others
duration	0.052897	.0577	0.004084	0.0008	0.0014
Protocol_type	0.304063	.3046	0.003147	0.00013	0.00180
Service	0.570986	.5999	0.014867	0.00079	0.0341
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

For each column we sort descending in which we wil get ranking of features according to each attack class. We substitute the feature name with its number according to Fig. 2 and we wil obtain Table VI.

C. Data Preprocessing And Normalization:

Each record in the KDD99 data sets contains 41 features that belong to different range values as shown in the example in Table VII.

So normalization is needed to put al features in a homogenous values space. The folowing normalization measure is suggested:

$$f_i = normalized_{f_i} = I(X, Y_{f_i})\frac{f_i}{\max(F_i)}$$

- Where $I(X,Yf_i)$ is the mutual information contained feature f_i (or amount of information about attack contained in f_i)
- Max (F_i) is the maximum value of the feature fi in the complete KDD99 training dataset (4898431 connections) and is defined as the folowing:

Table VI: Mutual Information Based Importance Measure

Normal/attack	DoS/Others	R2L/Others	U2R/others	probe/others
5	5	5	6	35
23	23	3	5	3
3	3	23	1	5
6	6	33	3	27
36	24	24	14	33
12	12	6	10	40
24	36	10	13	30
2	32	32	17	24
32	2	37	32	34
37	37	12	33	4
33	33	22	23	25
35	35	1	24	37
31	34	2	12	38
34	31	35	16	29
29	30	34	2	41
30	38	39	41	23
39	29	38	36	28
38	39	40	35	32
26	25	4	37	31
25	26	11	40	6
4	4	29	18	2
1	1	30	29	12
41	41	26	34	36
40	40	41	39	1
10	27	25	25	39
16	10	36	4	26
28	28	31	30	10
19	22	28	38	13
13	16	17	27	8
27	19	13	26	22
17	13	16	9	16
8	17	14	11	19
22	8	19	31	17
18	11	27	19	11
15	14	9	28	14
11	18	18	8	18
14	15	15	22	7
9	9	8	7	15
7	7	7	15	9
20	20	20	20	20
21	21	21	21	21

Table VII: Features Values Differences

Featurename (*f I*)	Type	Values example	Max
duration	continous	565,255	58329
dst_bytes	Continous	125454,45454	1309937401
same_srv_rate	continous	0.23,	0.65 1
Protocol_type	Discrete,	text tcp, icmp, udp	×
Flag	Discrete,	text SF, REJ, S1	×

If f_i W_0 then we take the maximum over the complete data set.

If f_i =0 then Max(F_i)=1

If f_i is text like protocol_type, flag, service then we sort feature values in ascending order according to their probability distributions, then we assign serial numbers to them from 1 to last value. After then we normalize according to relation before. The next example shows how to normalize text features.

Ex. Normalizing Protocol_type Feature:

Table VIII shows how to normalize text values by sorting in descending order then numbering and then dividing by max value.

In previous example textual features are not only normalized between [0,1] but are given importance weight according to their distribution in the KDD connections so:

f_i is more important than $f_j \Leftrightarrow$ (YXI_{fi}) I(X, Y_{fj})

Table VIII: Protocol_type Feature Normalization

Protocol_type	records	sorting	Numbering	Dividing by max	f_i
ICMP	2833545	UDP	1	1/3	0.1013
TCP	1870598	TCP	2	2/3	0.2027
UDP	194288	ICMP	3	3/3	0.3041

Al KDD99 or other traffic wil be passed now through the "Normalization & Preprocessing Unit" as proposed in Fig 3. The output wil be considered the new KDD99 dataset.

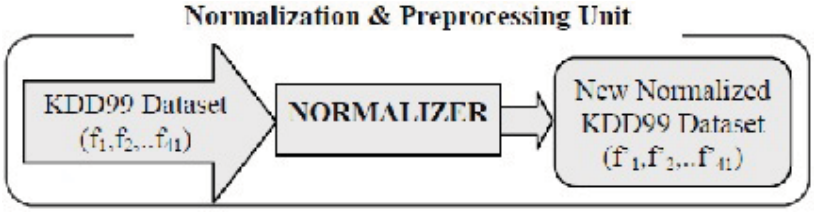

Fig. 3: Normalization & preprocessing unit

V. KNN Based Anomaly Ids:

In this section some anomaly models wil be introduced, al these and other models wil be evaluated according to performance measures.

A.lgorithm Performance Measures:

To evaluate how much the detection algorithm is good, the folowing measures must be calculated (indeed FPR, DR, and NDR are the most important ones):

- TP (True Positive): Number of connections that were correctly classified as Attack.
- TN (True Negative): Number of connections that were correctly classified as Normal.
- FP (False Positive): Number of normal connections that were classified as Attack.
- FN (False Negative): Number of attack connections that were classified as Normal.
- TPR = DR: Detection Rate= $\frac{TN}{TP + FN}$
- NDR : Detection Rate of new attacks

$$NDR = \frac{\text{nb of Newcorrectly detected attacks}}{\text{number of all newattack}}$$

Since the KDD99 testing dataset contains new attacks that were not included in the training data set. NDR can be said as "algorithm cleverness".

- TNR: True Negative Rate = $\frac{TN}{TN + FN}$
- CR :Classification Rate = $\frac{TP+TN}{\text{sizeof data set}}$
- FPR: False Positive Rate = $\frac{FP}{FP + TN}$
- FNR: False Negative Rate = $\frac{FN}{FN + TP}$

B.K-NN Detection Algorithm:

Fig. 4 shows how to use K-NN algorithm to classify the KDD99 testing dataset using the 1 0%KDD99 training dataset as reference.

```
Input KDD99 Testing datase U , KDD99 Training dataset V
For each connection u from U
    For each connection v from V
        calculate Dist=D(u,v)  // get distance between  U and V
    next
    sort V according to Dist
    select top k connections  // k-nearest neighbors;
    Classify u as the majority of classes of the selected connections
Next
```

Fig. 4: K-NN algorithm to classify the KDD99 dataset

Where the distance function is considered the Euclidian distance.

$$D(U,V) = \|U,V\| = \sqrt{\sum_{i=1}^{n} (f_i^U - f_i^v)^2} n = 41 \; features$$

C. Experimental Results For SF-KNN Algorithm:

Since this K-NN will be applied on a few selected features, it will be called SF-KNN:

The algorithm wil try to detect attacks in the test data. The test data contains a total of 38 attack types; of which 14 are new (i.e. the algorithms were not trained with instances of these attack types).

Fig. 5 shows a block representation of the whole detection operation using SF-KNN

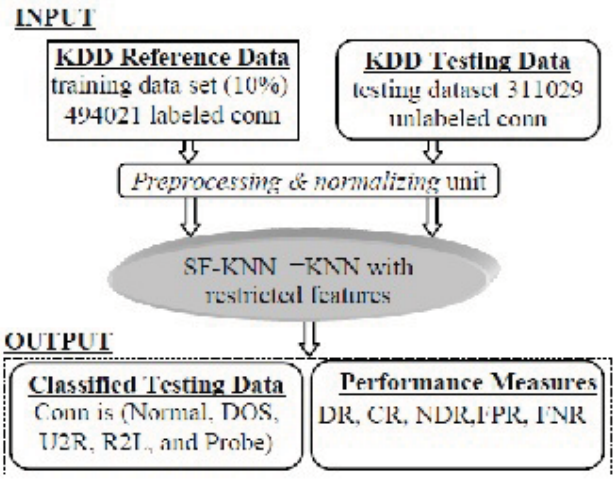

Fig. 5: SF-KNN detection operation

Four experiments have been applied using random samples from the training dataset, and then the parameters found have been applied to the testing data set. Four experiments have been executed.

1) Experiment 1: Using Top 8 Features to Select Best K for SF-KNN

The K-NN algorithm was run for K=1, 3, 5, 7. For every value of K the algorithm was run to study the effect of network features on anomaly detection.

First runs were executed using the al 41 network connection features (blind run or not clever run).

Next runs we selected the most top 8 important features for every attack class as ranked according to mutual information measure in Table. Theses features are: duration, service, src_bytes, dst_bytes, count, rerror_rate, dst_host_srv_count, dst_host_dif_srv_rate.

The results of applying this algorithm on the testing data set emphasized the experiments results applied on the training dataset. These results are ilustrated in Table IX.

Table IX: Performance Measures for K=5 (BEST K)

CLASSES	Size	Detected	DR	NDR
Normal	60593	59384	98.00%	x
DOS	229853	222784	96.92%	3.98%
R2L	16347	1320	8.07%	0.13%
U2R	70	23	32.86%	32.26%
Probe	4166	3894	93.47%	88.54%
TOTAL	DR =91.05% NDR=9.97% CR=92.40%		FP=2% FN=8.95%	

2) Experiment 2: the efect of features selection on the detection accuracy:

The K-NN experiment was first executed on a random sample from the training data set giving best results for the top 8 features then the same experiment was performed on the test data set. The experiment performed on the test data set emphasized our decision to select top 8 features. The KDD99 test data set contains 311029 connection records and 70 U2R attack with 31 new attack connections (ps, xterm, sqlattack). Table X shows the results of this experiment.

Table X: the Effect of Features Selection on Detecting U2r Attack

U2R features	TP	FN	New	DR %	NDR %	New detected	New not deteced
Full 41features	35	35	14	50%	45.16%	ps, xterm	sqlattack
6,5,1,3,14,10 ,13,17,32,33	49	21	25	70%	80.65%	ps, xterm, sqlattack	-
6,5,1,3,14, 10,13,17	51	19	26	72.86%	83.87%	ps, xterm, sqlattack	-
6,5,1,3, 14,10,13	50	20	25	71.43%	80.65%	ps, xterm, sqlattac	-
6,5,1,3	27	43	14	38.57%	41.16%	ps, xterm	sqlattack

One can notice that using al parameters (features) of the problem wil not improve the solution but wil cause a noise to the problem of detection.

So one can conclude that not every feature of traffic data is relevant to the intrusion detection task and by consequent adding more features may cause negative effect to the detection task so it is very important to select the most important features instead of al ones.

3) Experiment 3: Probe features and detection system prototype:

This experiment tries to calculate the performance measures for PROBE attack. The 5-NN detection algorithm was executed on the KDD99 test data set which contains 311029 connection records and 4166 probe attack with 1789 new attack connections (mscan, saint). Firstly, the experiment was performed on a random sample of the training data set founding that using top 4 features is better that using top 8 features, this was realy approved when This experiment was reapplied on the testing data set as ilustrated in Table XI.

Table XI: Probe Attack Performance Measures

PROBE features	TP	FN	new attacks	DR	NDR	New detected
35,3,5,27, 33,40,30,24	3252	914	1320	78.06%	73.78%	all
35,3,5,27	4146	20	1776	99.52%	99.27%	all

One can see that using 4 features is better than 8 features to detect probe attacks. Besides a high detection rate for old and new attacks was obtained.

4) Experiment4: R2L features and detection system prototype:

The 5-NN detection algorithm was executed on the KDD99 testing dataset which contains 311029 connection records and 16347 R2L attacks with 10354 new attack connections (named, sendmail, xlock, xsnoop, httptunnel, snmpgetattack, snmpguess and worm). Table XII shows the experiment results.

Table XII: U2R Attack Performance Measures

R2L features	TP	FN	New attacks	DR	NDR	New detected
5,3,23,33, 24,6,10,32	1306	15041	20	8.68%	0.19%	all

From the experiment one can see that the DR and NDR is very low, the explanation of this is that when Remote hacker tries to attack and hack a local user, he wil create a legal environment so the attacker seems as a legal local user. Fig. 6 is an example of normal and U2R connections, one can see that they are very close to each other and by consequent this R2L connection is very hard to be detected.

Feature name	Normal	R2L (snmpgetattack)
duration	0	0
protocol_type	Udp	Udp
service	Private	Private
.		
.		
count	1	2
srv_count	1	2
same_srv_rate	1	1
diff_srv_rate		
.	.	.
.		

Fig. 6: Normal and R2L attacks are so similar

D. SF-5NN Anomaly IDS Architecture:

According to results of experiment 1, 2, 3, and 4 that are summarized in Table XIII one can propose the folowing anomaly intrusion detection system architecture (Fig. 7) which is based on 5-NN algorithm

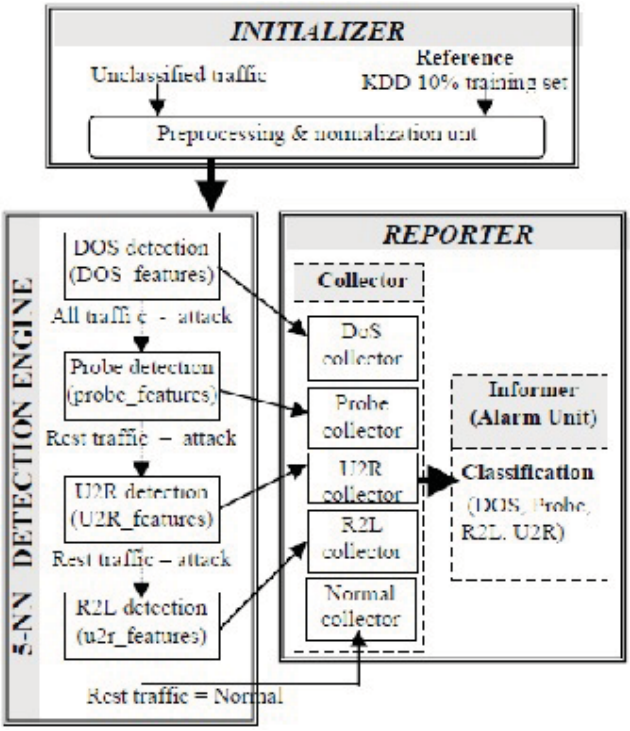

Fig. 7: Anomaly IDS based on 5-NN algorithm

- System Purpose: The system analyses one single connection or a LOG resultant of sniffing network traffic by TCPDUMP and then tel for each connection in the record if there is an attack. The system can also be caled as LOG ANALYSER.
- System Description: Our IDS consists of three main units:
  - Initializer: this unit prepares captured network traffic to the Detection Engine Unit where it wil process and normalize data as seen before in paragraph IV-D.
  - 5-NN detection engine: this unit consists of four subunits to detect attack classes as ilustrated in Table XIII

Table XIII: Connection Features Adopted in Our Ids

DoS detection subunit	Probe detection Subunit	U2R detection Subunit	R2L detection Subunit
1,3,5,6,23,27,33,35	35,3,5,27	6,5,1,3,14,10,13,17	5,3,23,33, 24,6,10,32

DoS subunit analyses every connection record to detect attacks according to DOS_features, the result of the analyses (DOS, U2R, R2L, and Probe) wil be send to the appropriate subunit in the colector. Normal classified records wil not send to Normal collector.

Next step is to send al normal classified DOS records to Probe Unit to detect more attacks according probe features, the result wil be treated similarly to previous step. Rest traffic wil be send to U2R unit then R2L detection unit. The rest traffic resultant of R2L detection unit wil be classified by our system as normal.

- Reporter: consists of two main units, the colector and the informer. The colector colects each connection according to their classification and passes them to the informer which sends alarm to the user when the connection is an attack, the attack may be labeled according to its classification. Of course this classification may not be accurate as explained.
  - False Positive: The connection is classified as an attack while it is normal so the IDS assessment is not correct.
  - True positive with correct classification: The connection is an attack and classified properly. For example the connection is U2R and classified as U2R so the IDS assessment is perfect.
  - True Positive but miss classification: The connection is an attack and classified as another type of attack For example, the connection is U2R but classified as PROBE so the IDS assessment is not good.

E. Proposed IDS Performance Measures:

In previous each module was tested separately (DOS, U2R, R2L, and Probe). Now we wil run our anomaly IDS described in Table XIII and Fig. 7 on the KDD99 test dataset. Table XIV shows the results that have been found. Also Table XV presents a comparison between using common features for al attacks classes and using selected features for each class.

Table XIV: SF-KNN Anomaly Ids Performance Mesures

CLASSES	Size	Detected	DR	NDR
Normal	60593	59384	98.00%	x
DOS	229853	222978	97.01%	5.86%
R2L	16347	1306	7.99%	0.19%
U2R	70	54	77.14%	87.10%
Probe	4166	4155	99.74%	99.55%
TOTAL	DR =91.24% NDR=11.81% CR=92.56%		FP=2% FNR=8.76%	

Table XV: Comparison of Experiment1results and SF-5nn Anomaly Ids

CLASSES	Top 8 common features	SF-5NNAnomaly IDS
NORMAL	98.00%	98.00%
DOS	96.92%	97.01%
PROBE	93.47%	99.74%
R2L	8.07%	7.99%
U2R	32.86%	77.14%

One can see that detection rates when using different selected features for each attack class is better than using common selected features for al classes. Obviously, This procedure improved Probe and U2R attacks. The proposed "SF-5NN anomaly IDS" has detected al types of known and unknown attacks except "phf" attack from the known group and "mailbomb" from the unknown attack.

F. Sus-knn Anomal Ids:

SUS KNN or SUSpect with KNN is the same of our 5NN based IDS with folowing difference:

- K-NN: compares the test connection with the K nearest neighbors in the training data and the decision is taken according the majority
- SUS-KNN: compares the test connection with the K nearest neighbors in the training set, if there is at least one attack the connection wil be classified as an attack.

This method wil cause more false alarms so it may be restricted on serious attacks like U2R and R2L. Table XVI compares between the two anomaly IDSs.

Table XVI: Comparison Between Sf-5nn and Sus-5nn

CLASSES	SF-5NN		SUS-5NN	
	DR %	NDR %	DR%	NDR %
NORMAL	98	-	95.43	-
DOS	97.01	5.86	97.14	7.77
R2L	7.99	0.19	10.08	0.81
U2R	77.14	87.10	91.43	96.77
PROBE	99.74	99.55	99.86	99.78
	TOTAL %			
TPR	91.24	91.50		
TNR	98.00	98.35		
CR	92.56	92.84		
FPR	2.00	4.58		
FNR	8.76	8.50		

A seen in the table, the detection rate of U2R attack was seriously increased when adopting SUS-5NN strategy. A little improvement was done on detecting R2L attack and others, but the false positive rate was sorrowfuly doubled.

G. Advantages and Disadvant. of the SF-5NN Anomaly. IDS:

Advantages

- Since it is essentialy based on 5NN algorithm it Easy to establish
- It provides us with an excelent detection rate for Normal connections and PROBE, DOS attacks.
- It can be easily converted to SUS-KNN which in turn improves the detection rates but increases the false positive rate.

Disadvantages

- Time consuming, since each unclassified connection must be compared with 494021 labeled connection to get the 5-NN.
- The detection rate of R2L attack is low.
- Needs to normalization and preprocessing work.
- It wil be biased towards training set.

VI. K-means Clustering Based Anomaly Ids:

Since, In KNN based IDSs, each connection must be compared with 494021 records to determine its classification, it can be proposed to use k-means clustering algorithm with K-NN to reduce the complexity (see Fig.8). For this there are two propositions.

- Method1: partitioning the reference training data (KDD99 10% train set = 494021 record) into k primitive clusters, hoping that homogenous connections wil be grouped and compacted together.

The purpose of using k-means algorithm is to substitute the huge training data with clusters centroïds which wil decrease dramaticaly the K-NN complexity. So instead of comparing an unclassified connection with 494021 records to find the K-NN, we compare it with clusters centroïds. It is obvious that for more number of clusters one wil have better classification, but more computation. Fig. 9 shows the detection process using SF-KNN and K-means algorithm by clustering the training data set. An experiment was executed considering this method and the results were not so bad (Table XVII).

- Method2: the same technique can be applied by clustering the unclassified dataset (logfile = captured traffic) into N clusters and then each cluster centroïd must be classified according to its K nearest neighbors in the training dataset. If the centroïd is classified as an attack then al the cluster members wil be considered as an attack. Fig. 10 shows the detection process using SFKNN and K-means algorithm by clustering the testing data set. An experiment was executed considering this method. The results found were better than method 1 (Table XVII).

Table XVII: Detection Rates Using "Sf-5nn Anomaly Ids" with Clustering Training and Testing Datasets

Clusters	CR (training set)	CR (testing set)	reduction in execution time
10	74.03 %	91.11 %	0.00002024 *Time SF-KNN
34	81.56 %	91.27 %	0.00006882 *Time SF-KNN
118	81.72 %	91.34 %	0.00023886 *Time SF-KNN

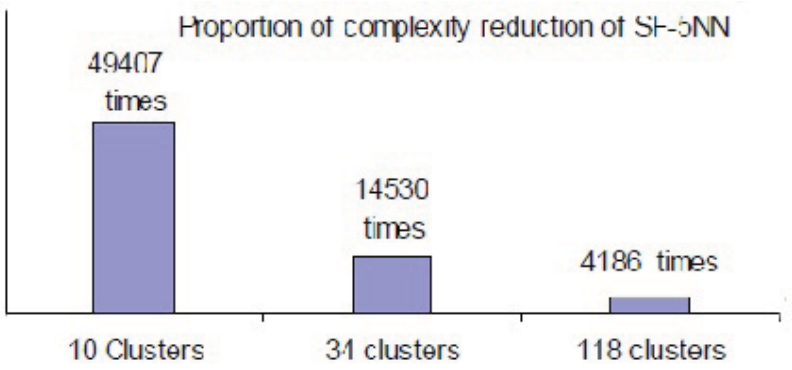

Fig. 8: Clustering leads to reduction in complexity reference to SF-5NN

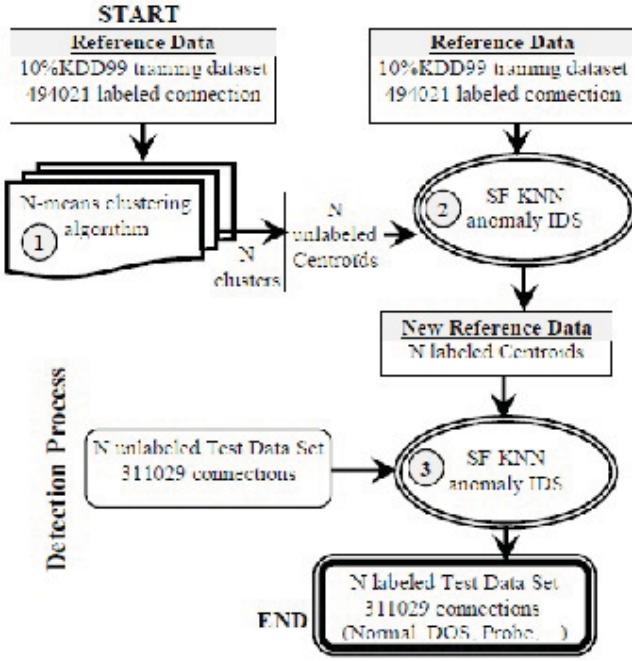

Fig. 9: Anomaly detection using k-means algorithm to reduce the SF-KNN complexity - Clustering the reference (training) dataset

The distance function adopted in both methods uses only top common 8 features (duration, service, src_bytes, dst_bytes, count, rerror_rate, dst_host_srv_count, dst_host_diff_srv_rate).

VII. Comparing Approches and Gathering All Anomaly Models:

Comparing Approaches:

Table XVIII and Fig. 11 show a comparison between our approaches and others (Chi-Ho Tsang, Sam Kwong, 2005):

Fig. 11 shows that our approaches are of the best approaches especialy for detecting U2R and Probe attacks.

Also our approach is much better than other 5-NN based algorithm.

A draw back of our K-NN based systems is that they are slow (al K-NN based algorithm are slow), so it is better to use them as LOG ANALYSER.

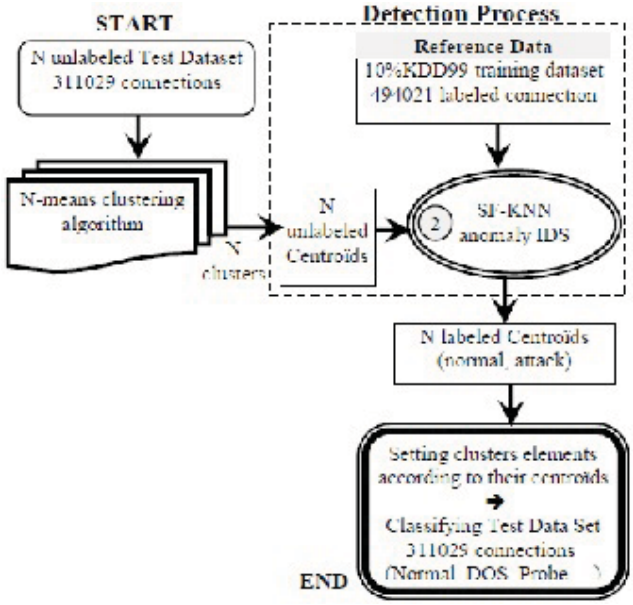

Fig. 10: Anomaly detection using k-means algorithm to reduce the SF-KNN complexity - Clustering the Testing (log file) dataset

Table XVIII: Comparing Our Approach to Design Anomaly Ids with Other's

Our Anomaly IDS s						Other's IDSs									
CLASS	Top 8 common features	SF-5NN ANOMALY IDS	SUS-5NN ANOMALY IDS	(118 CLUSTER of trainingset)	(118 CLUSTER of testing set)	Other 5-NN	Runner up of KDD [10]	RSS-DSS [11]	MOGF-IDS [9]	PN rule [12]	CTree [5]	C5	C4.5	SVM	Cup winner [13]
NORMAL	98.00	98	95.43	CR	CR	95.89	99.4	96.5	98.36	99.5	92.78	99.5	98.38	97.99	99.5
DOS	96.92	97.01	97.14			97.00	97.5	99.7	97.20	96.9	98.91	97.1	96.99	97.56	97.10
R2L	8.07	7.99	10.08	=	=	6.90	7.3	31.2	11.01	7.3	7.41	8.4	1.45	3.55	8.40
U2R	32.86	77.14	91.43			14.91	11.8	76.3	15.79	11.8	88.13	13.2	14.47	10.09	13.20
PROBE	93.47	99.74	99.86	81.72	91.34	81.61	84.5	86.8	88.6	84.5	50.35	83.3	81.88	81.61	83.30

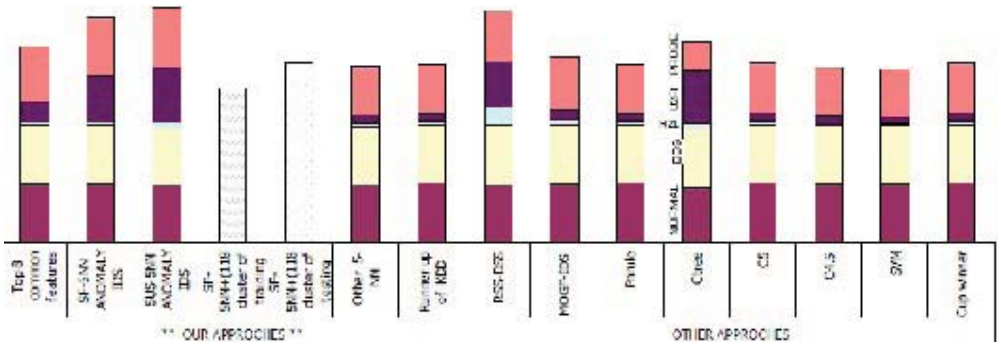

Fig. 11: Detection Rate Comparison

Gathering All Anomaly Models:

Al previous approaches (our approaches + others) can be programmed into one package which may be caled "Anomaly IDS package" as shown in Fig.12a. Then it is up to the user to select one anomaly detection method

Then to position the "Anomaly IDS package", there are two possibilities:

- Standalone: put the package immediately on the network. Of course we need firstly:
  - sniffer or TCPDUMP software
  - tcpdump_to_kdd99 function to convert formats.

- After a signature based IDS like SNORT. In this Case SNORT or other wil alarm for bad signatures and then the resident traffic which is classified as normal wil be send to tcpdump_to_kdd99 transformer an then processed by "Anomaly IDS package".
  Fig. 12b. Shows an ilustration of these two possibilities

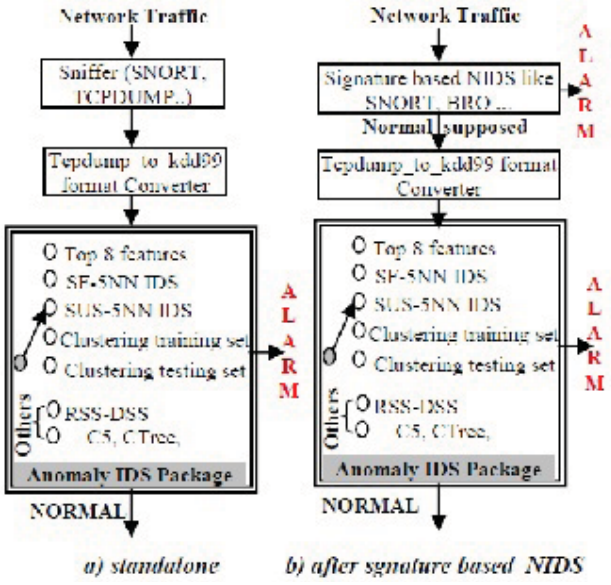

Fig. 12: Anomaly IDS Package" and its positioning

VIII. Comclusion and Future Work:

A. Conclusion:

By applying the information theory measures like entropy and mutual information, the 41 connection features were ranked after normalization process according to each attack class. This ranking alows decreasing the computing complexity by selecting the most important features for each attack class. Features selection proved that they dramaticaly decreased the detection speed without affecting the detection rates. Besides it was proved that running some detection models like SF-5NN and SUS-5NN using few selected features is more effective than running it with the ful 41 features. This is because some features are not relevant to the detection task and may cause noise. Later two detection engines based on best selected features and K-NN algorithm were proposed, SF-KNN and SUS-KNN. These engines provided a good classification rates equals to 92.56%, 92.84% and false positive rates 2%, 4.52% respectively.

The proposed models were better than many other approaches like traditional 5-NN, C4.5, C5 and cup winner especially in detecting dangerous attacks like U2R and R2L. Besides, these models could detect all types of known and unknown attacks types except "phf" and "mailbomb" attacks.

The drawback of SF-KNN and SUS-KNN is their computation complexity. K-means clustering algorithm was used to cluster either the training dataset or the testing dataset The detection rates found were 81.56% when clustering the reference dataset (training) into 34 clusters and 91.11% when clustering the testing dataset into 10 clusters. The detection rates are less than before but when clustering the detection times were reduced by 14530, 49407 times for training datasets and testing dataset respectively. In spite of this decrementing in processing time, the previous models still ineffective if used in real-time but are good if used as off-line LOG analyzers.

A package called "Anomaly IDS package" was proposed to detect anomaly in network traffic. This package gathers all IDSs proposed in the paper and others. It functions standalone or with a traditional signature based IDS like SNORT. Finally, one can see that all previous models are modularized so that new learning algorithms are easy to be added in and tested or even inserted as a new module. This well suits the dynamics in the research world.

B. Future Work:

Many recommendations can be proposed for the future work like:

- Put and test all previous models in the real world.
- To make the previous models as general as possible, the training data set must be as variant as much as possible.

A proposition may be:

- o Besides the KDD99, training using the Supelec TCPdump dataset (30GB size, TCP: 1173654; UDP: 3254160; ICMP: 224245)
- o Train and test the previous models on web server logs which consist has a lot of URL request information. Those logs have more text content than the binary logs captured by TCPDUMP.
- Building a testing system similar to the simulated network that generated KDD99 data set. The testing system can be used to:
- o generate new forms of attacks extracted from different distributions
- o Test the previous LOG ANALYSERS models
- Since the detection engines in the paper are modular. One can propose importing other's successful approaches and insert them as new modules in the detection engine.
- all these models need a lot of time in the analysing or training process, one can propose using more developed infrastructure. A suggestion of this infrastructure is using Microsoft Computing Cluster. This technique enables to group many computers as one cluster and then the training process could be started on this cluster

REFERENCES

Agarwal, R., M.V. Joshi, 2000. "PNrule: A New Framework for Learning classifier Models in Data Mining," Department of Computer Science, University of Minnesota, Report No. RC-21719.

Athanasios Papoulis and S. Unnikrishna Pillai, 2002. "Probability, Random Variables and stochastic Processes", Book.

Book written by Athanasios Papoulis and S. Unnikrishna Pillai, 2002. "Probability, Random Variables and stochastic Processes.

Chi-Ho Tsang, Sam Kwong, 2005. Hanli Wang1, Anomaly Intrusion Detection using Multi-Objective Genetic Fuzzy System and Agentbased Evolutionary Computation Framework, Proceedings of the Fifth IEEE International Conference on Data Mining.

Elkan, C., 2000. Results of the KDD'99 classifier learning, ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, Boston, MA, 1(2): 63-64.

http://kdd.ics.uci.edu/databases/ kddcup99/kddcup99.html.

Levin, I., 2000. KDD-99 Classifier Learning Contest LLSoft's Results, Overview. SIGKDD Exulorations. ACMSIGKDD, 1(2): 67-75.

Liwei (Vivian) Kuang, 2007. DNIDS: A Dependable Network Intrusion Detection System Using the CSI-KNN Algorithm,Master of Science, Queen's University Kingston, Ontario, Canada, September.

Marimuthu, A. and Dr. A. Shanmugam, 2008. "Intelligent Progression for anomaly Intrusion detection", 6th International Symposium on Applied Machine Intelligence and informatics, SAMI.

Melanie J. Middlemiss and Grant Dick, 2003. "Weighted Feature Extraction using a Genetic Algorithm for Intrusion Detection", 3: 1699- 1675, Evolutionary Computation, CEC '03.

Prerau, M.J. and E. Eskin, Unsupervised anomaly detection using an optimized K-nearest neighbors algorithm. Master's thesis. Available at http://www.music.columbia.edu/~mike/ publications/thesis.pdf.

Qu, X., S. Hariri, M. Yousif, "An Efficient Network Intrusion Detection Method Based on Information Theory and Genetic Algorithm", Proceedings of the 24th IEEE International Performance Computing and Communications

Song, D., M.1. Heywood, A.N. Zincir-Heywood, 2005. Training Genetic Programming on Half a Million Patterns: An Example from Anomaly Detection. IEEE Transactions on Evolutionary Computation, 9(3).