HawkEye Solutions: Expectation Maximization based Network Intrusion Detection System

I. Mukhopadhyay
Institute of Engineering
& Management
Y-12 Salt Lake Electronics
Complex
Kolkata - 700091

M. Chakraborty
Institute of Engineering
& Management
Y-12 Salt Lake Electronics
Complex
Kolkata - 700091

ABSTRACT
This paper projects a novel Network Intrusion Detection System (NIDS) known as HawkEye Solutions that detects abnormal Internet Protocol (IP) packets. An NIDS is a computer-based information system designed to collect information about malicious activities in a set of targeted IT resources, analyze the information and respond according to some predefined security policy. Authors here present the basic building blocks of the IDS that include mechanisms for carrying out TCP port scans, Traceroute scan, which in association with the ping scan can monitor network health. Finally the implementation of a Packet Sniffer provides generic level opportunity to detect various types of attacks, based on packet analyzing on a real-time basis. The authors have also proposed and implemented a novel Expectation Maximization based intrusion detection algorithm called EMID. The implementation results in Matlab are presented and discussed. The authors endeavor to integrate EMID with HawkEye Solutions as a future work.

General Terms
Security

Keywords
Intrusion Detection System (IDS), Intrusion Prevention System (IPS), Host-based IDS, Network-based IDS, HawkEye Solutions, Expectation Maximization, Bayesian Classifier, EMID.

1. INTRODUCTION
In recent years, Intrusion Detection Systems find tremendous importance in the field of detection technology and is used as a countermeasure to preserve data integrity and system availability during an intrusion. Sometimes an intruder attempts to break into an information system or performs an action not legally allowed. This activity is known as an intrusion. Intruders can be divided into two groups, external and internal. The former refers to those who do not have authorized access to the system and who attack by using various penetration techniques. The latter refers to those with access permission who wish to perform unauthorized activities. Intrusion techniques may include exploiting software bugs and system misconfigurations, password cracking, sniffing unsecured traffic, or exploiting the design flaw of specific protocols.

The Internet is a global network of interconnected computers enabling users to share information along multiple channels. A computer connected to the Internet is able to access information from a vast array of available servers and other computers by moving information from them to former computer's local memory. Common uses of the Internet are Email, World Wide Web, remote access, collaboration, streaming media, file sharing. But nowadays malfunctions on the Web are increasing. There are computer investment frauds, cyber crimes, financial crimes, phishing scams, chatting (masquerading) and crimes associated which share trading on Web.

Network Security consists of the provisions made in an underlined computer network infrastructure and policies adopted by the Network Administrator to protect the network and network accessible resources from unauthorized access, consistent and continuous monitoring and measurement of its effectiveness combined together. Computer forensics is a branch of forensic science pertaining to legal evidence found in computers and digital storage mediums. Computer forensics is also known as digital forensics. The goal of computer forensics is to explain the current state of digital artifacts. The term digital artifact may include a computer system, a storage medium (such as hard disk or CDROM), an electronic document (and email message or JPEG message) or even a sequence of packets moving over a computer network [1], [2].

Different techniques of computer forensics are used for

- Recovering data in the event of a hardware or software failures.
- Analyzing a computer system after a break in, for example, determining how the attacker gained access and what the attacker did.
- Gaining information about how computer systems work for the purpose of debugging, performance optimization, or reverse-engineering.

However complete prevention of breaches of security is unrealistic. The authors have highlighted the characteristics of abnormal Internet Protocol (IP) packets in this paper and have proposed a novel approach to intrusion detection by combining two machine learning algorithms – Expectation Maximization [3] algorithm followed by Bayesian Classification [4]. The algorithm is known as EMID: A Novel Expectation Maximization based Intrusion Detection Algorithm. Abnormal packets are defined as those packets that violate the IP protocol standards. Abnormal packets may be generated through benign means, such as a malfunctioning router, but they are usually

specially crafted by attackers. The abnormality is often introduced into the packet purposely so that the packet may avoid being blocked by a firewall or intrusion detection system. The effort was to design a basic Intrusion Detection System (IDS) called HawkEye Solutions that can be implemented even by naïve users who do not have the background for understanding the details. The level of abstraction for the user is kept high so as to provide a summary regarding abnormal packets (if any) and also try and explain the cause of such abnormal behavior.

The organization of the paper is as follows. After the introduction in section 1, section 2 talks about the architecture of HawkEye Solutions. The working principle of HawkEye Solutions is discussed in section 3. This is followed by the theoretical background of EMID, the algorithm for IDS in section 4. Real time implementation results of HawkEye Solutions are provided in section 5. Section 6 concludes the paper with some highlights on future works.

2. ARCHITECTURE OF HAWKEYE SOLUTION

HawkEye Solutions is a libpcap-based [5] packet sniffer and logger that can be used as a lightweight NIDS. It features rule based logging to perform content pattern matching and detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, and much more. HawkEye Solutions has real-time alerting capability, with alerts being sent to syslog, popup messages, or a separate ‘‘alert’’ file. HawkEye Solutions is configured using command line switches. The detection engine is programmed using a simple language that describes per packet tests and actions. Ease of use simplifies and expedites the development of new exploit detection rules.

The block diagram representation of the architecture of HawkEye Solutions is shown in figure1.

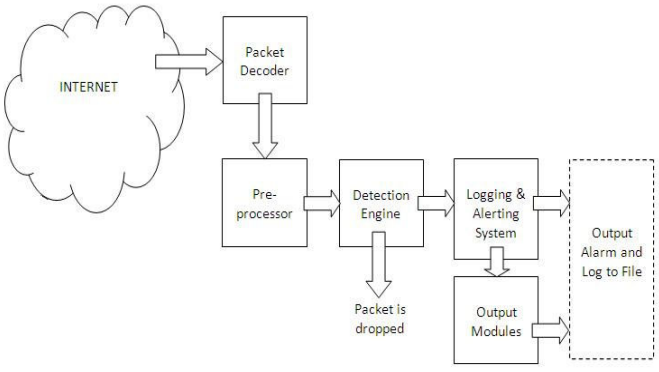

Figure 1 Architecture of HawkEye Solutions

HawkEye Solution’s architecture is focused on performance, simplicity, and flexibility. There are three primary subsystems that make up HawkEye: the packet decoder, the detection engine, and the logging and alerting subsystem. These subsystems ride on top of the libpcap promiscuous packet sniffing library, which provides a portable packet sniffing and filtering capability. Program configuration, rules parsing, and data structure generation takes place before the sniffer section is initialized, keeping the amount of per packet processing to the minimum required to achieve the base program functionality.

The libpcap library used in HawkEye Solutions can be used to read, record, inject and in general deal with network packets at a higher level than raw sockets. Essentially libpcap can be used to easily collect up or manipulate packets. The Packet Capture library provides a high level interface to packet capture systems. All packets on the network, even those destined for other hosts, are accessible through this mechanism.

HawkEye Solutions is cosmetically similar to tcpdump but is more focused on the security applications of packet sniffing. The most important most features possessed by HawkEye Solutions unlike tcpdump are packet payload inspection. HawkEye decodes the application layer of a packet and can be given rules to collect traffic that has specific data contained within its application layer. This allows HawkEye Solutions to detect many types of hostile activity, including buffer overflows, CGI scans, or any other data in the packet payload that can be characterized in a unique detection fingerprint.

Another advantage of HawkEye Solutions is that its decoded output display is somewhat more user friendly than tcpdump’s output. HawkEye Solutions can utilize its flexible rule set to perform additional functions, such as searching out and recording only those packets that have their TCP flags set a particular way.

3. WORKING PRINCIPLE OF HAWKEYE SOLUTION

This section deals with the working principle of HawkEye Solutions. The various steps followed by HawkEye Solutions are as follows:

1. An event record is created. This occurs when an action happens; such as packets of data transmitting in the network or even a file is opened or a program is executed like the text editor like Microsoft Word. The record is written into a file that is usually protected by the operating system trusted computing base.
2. The target agent transmits the file to the command console. This happens at predetermined time intervals over a secure connection.
3. The detection engine, configured to match patterns of misuse, processes the file.
4. A log is created that becomes the data archive for all the raw data that will be used in prosecution.
5. An alert is generated. When a predefined pattern is recognized, such as access to a mission critical file, an alert is forwarded to a number of various subsystems for notification, response, and storage.
6. The security flag / message are sent i.e. notified.
7. A response is generated. The response subsystem matches alerts to predefined responses or can take response commands from the security officer. Responses include

reconfiguring the system, shutting down a target, logging off a user, or disabling an account.

8. The alert is stored. The storage is usually in the form of a database. Some systems store statistical data as well as alerts.
9. The raw data is transferred to a raw data archive. This archive is cleared periodically to reduce the amount of disk space used.
10. Reports are generated. Reports can be a summary of the alert activity.
11. Data forensics is used to locate long-term trends and behavior is analyzed using both the stored data in the database and the raw event log archive.

The flow diagram of the steps discussed above is shown in figure 2. The lifecycle of an event recorded through the proposed architecture is advantageous as everything happens in real-time. The disadvantage is that the end users suffer from system performance degradation.

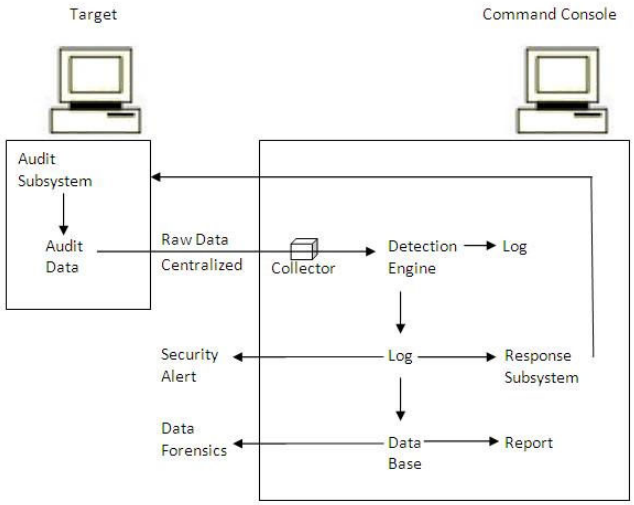

Figure 2 Flow diagram of working principle of HawkEye Solutions

4. EMID: ALGORITHIM FOR NIDS

The task of developing EMID (a network intrusion detection system) is to analyze the authentication records and separate genuine and fraudulent authentication attempts for each user account in a system. To achieve this, the authors have combined two machine learning algorithms – Expectation Maximization (EM) algorithm followed by Bayesian Classification (BC). The EM algorithm uses a maximum likelihood type parameter estimation approach and is ideally suited for cases in which the available data set is incomplete. It is an iterative procedure which under given certain conditions converges to values at a local or global maximum of the likelihood function. EMID performs the analysis of the authentication data in order to separate genuine and fraudulent authentication attempts for each user account in the system.

The authors have adopted a finite mixture model for our IDS which assume that the authentication data arises from two or more groups with known distributional forms but different, unknown, parameters. The following probabilistic model is used to describe this situation:

$$p(X \mid \theta) = \sum_{k=1}^{K} \alpha_k \; p_k (X \mid \theta) \tag{1}$$

where the unknown parameters $\theta = (\alpha_1, .. \alpha_k, \theta_1, .. \theta_k)$

are such that $\sum_{k=1}^{K} p_k$ & each p_k is a probability density function parameterized by θ_k. Thus, K component densities are mixed together using K mixing coefficients α_k. Our system assumes multivariate Gaussian component distributions so the *k*-th component can be fully specified using parameters μ_k (mean value) and $\sum_k$ (covariance), i.e. $\theta_k = (\mu_k, \sum_k)$ X represents the set of N observed samples from this mixture. Each sample X from this set is a multi-dimensional authentication data vector $X = [x_1, x_2, ...x_D]$ where D represents dimension of the sample, i.e. dimension of K multivariate Gaussian distributions considered. The task is to classify those data vectors to K different classes adopting the Bayes classification rule to assign the particular sample X to a class with the highest posterior probability $P(\omega_k \mid X)$. This is a probability that a sample vector X belongs to a class ω_k and can be computed using Bayes formula [4]:

$$P(\omega_k \mid X) = \frac{P(X \mid \omega_k) P(\omega_k)}{P(X)} \tag{2}$$

To calculate posterior probability the prior probability

$P(\omega_k)$ which specifies the initial probability - an observed feature vector belongs to a class ω_k as well as probability density function $P(X \mid \omega_k)$ must be known. This is the probability for data X given the class ω_k also known as class-conditional probability. $P(X)$ represents the sum of posterior probabilities across all classes:

$$P(X) = \sum_{k=1}^{K} P(X \mid \omega_k) P(\omega_k) \tag{3}$$

Its value is therefore same for all posterior probabilities considered. As such it can be neglected during $P(\omega_k \mid X)$ the classification and the classification rule can therefore be simplified to the assignment of vector X to a class ω_k if:

$$P(\omega_k \mid X) > P(w_j \mid X) for j = 1,2,..K \,\&\, j \neq k \tag{4}$$

It is known that this classification rule yields a classifier with the minimum probability of error [6]. The major problem in applying the Bayesian classifier is the estimation of class-conditional probability density function $P(X \mid \omega_k)$ as parameters of that class need to be known or if not known estimated from the measured data set using some of the available parameter estimation methods. Method used in this work employs the Expectation-Maximization (EM) algorithm [7]. EM is a general technique of estimating features of a given data set applicable when analyzed data are incomplete or have missing values. It is an iterative algorithm, where there are two steps for each iteration – expectation (E) step and maximization (M) step.

The following notation is useful for the further discussion about the EM algorithm:

X : set of all available features of all observed samples;

Y : set of all unknown features of all observed samples;

θ^i *:* estimate of distribution parameters at the *i*-th stage of the algorithm iteration;

θ: variable for the new estimate describing the (full) distribution.

The aim of the EM algorithm is to maximize the expected log-likelihood of the model parameters set θ given the joint distribution of the observed data X and the missing data Y. For a set of independent samples X, drawn from a single distribution described with the set of parameters θ, the likelihood function can be thought of as a function of the parameters θ where data X is fixed, i.e. it gives the likelihood L of the data X given distribution parameters θ.

$$L(\theta \mid X) = p(X \mid \theta) = \prod_{n=1}^{N} p(X_n \mid \theta) \tag{5}$$

In order to find the set of parameters θ that maximizes the likelihood, the usual approach is to avoid direct likelihood maximization and to try to maximize the logarithm of the likelihood function, i.e. log-likelihood function, which is analytically easier to handle since:

$$\ln L(\theta \mid X) = \ln p(X \mid \theta) = \sum_{n=1}^{N} \ln p(X_n \mid \theta) \tag{6}$$

The unobserved part of data set Y for this system can be considered as knowledge of which component *k* produced each sample X_n. For each sample X_n there is a binary vector $Y_n = \{y_{n,1},.....y_{n,K}\}$ where $y_{n,k} = 1$ if the sample was produced by the component *k* or zero otherwise. The log-likelihood function to be maximized by the EM algorithm considers the joint distribution of the observed data X and the missing data set Y and it can be shown that [8]:

$$\ln L(\theta \mid X, Y) \approx \sum_{n=1}^{N} \sum_{k=1}^{K} y_{n,k} \ln(\alpha_k p(X_n \mid \theta_k)) \tag{7}$$

In the E-step, the EM algorithm calculates the conditional expectation of the complete data log-likelihood function, given X and the current estimate $\theta = \theta^i$ of the parameters. It therefore evaluates the function:

$$Q(\theta \mid \theta) = E[\ln L(\theta \mid X, Y) \mid X, \theta] = \ln L(X, W \mid \theta) \tag{8}$$

where W represents the conditional expectation of Y with respect to set of observed data and parameter θ, i.e.,

$W = E[Y \mid X, \theta]$. Elements of W are defined as:

$$w_{n,k} = E[y_{n,k} \mid X, \theta^i] = P(y_{n,k} = 1 \mid X_n, \theta^i) \tag{9}$$

i.e. those are the probabilities that particular data sample X_n is produced by component k. The above probability can be calculated using Bayes law [8]:

$$w_{n,k} = \frac{\alpha^i{}_k \, p(X_n \mid k, \theta^i)}{\sum_{j=1}^{K} \alpha^i{}_j \, p(X_n \mid j, \theta^i)} \tag{10}$$

Where α_k^i represent the prior probability of class k at iteration step i. The task on M-step is to maximize $Q(\theta, \theta^i)$ with respect to θ and set:

$$\theta^{i+1} \leftarrow \arg_{\theta} \max Q(\theta;\theta_i) \quad (11)$$

Applying the M-step to the problem of estimating distribution parameters for K-component, D-dimensional Gaussian mixture with arbitrary covariance matrices, results in the following set of iteration equations [6]:

$$\alpha_k^{i+1} = \frac{1}{N}\sum_{n=1}^{N} w_{n,k} \quad (12)$$

$$\mu_k^{i+1} = \frac{\sum_{n=1}^{N} X_n w_{n,k}}{\sum_{n=1}^{N} w_{n,k}} \quad (13)$$

$$\sum_{k}^{i+1} = \frac{\sum_{n=1}^{N} w_{n,k}(X_n - \mu_k^{i+1})(X_n + \mu_k^{i+1})^T}{\sum_{n=1}^{N} w_{n,k}} \quad (14)$$

The E- and M-steps of the algorithm are repeated until a convergence criterion is met. The convergence criterion is usually chosen so that EM is interrupted when change in values evaluated at each algorithm iteration falls below a certain threshold value. Two possible criterions considered in literature are change of the estimation parameter set θ^{i+1} compared to the previous value of θ^{i} [6]:

$$\left\|\theta^{i+1} - \theta^{i}\right\| \le \varepsilon \quad (15)$$

or the amount of change of log-likelihood expectation function between two iteration of the algorithm [9], i.e.:

$$Q(\theta^{i+1},\theta^{i}) - Q(\theta^{i},\theta^{i-1}) \le T \quad (16)$$

For suitable threshold values of ε and T .

5. IMPLEMENTATION RESULTS

This section provides the real time implementation results of HawkEye Solutions for trace route scan and abnormal packet detection through its packet sniffing utility.

Figure 3 shows the screenshot of trace route scan. On selecting the Trace Route Scan option, a textbox appears on the right hand panel that requests the user to enter the IP address or URL of the destination to be traced. The output consists of 3 columns corresponding to each router or hop. Each of the 3 columns is a response from the concerned router in terms of how long it took (each hop is tested 3 times). The result of the scan is shown in the output text box and is automatically saved into the log file ScanTrace.txt

Figure 4 shows the screenshot of packet sniffing utility. On selecting the Packet Sniffer option and on clicking the Start button, the sniffing of packets starts with the packet details and data of each packet shown instantaneously. The information shown in the figure includes the details of Ethernet header, IP header and TCP/UDP header [10].

The packet sniffer also detects the abnormal packets (if any) and the cause for the abnormality for individual packets. The screenshot of the result is shown in figure 5. These are displayed along with the total count of abnormal packets discovered up to that instant. The data contained in the packet is displayed in the hexadecimal and string format. The result of the scan is shown in the output text box and is automatically saved into the log file sniffer.txt.

EMID, the proposed IDS algorithm was simulated in MATLAB version: 7.5.0.342 (R2007b). Figure 6, figure 7 and figure 8 show the results of applying the proposed algorithm to authenticated data for three attributes both for fraudulent data as well as genuine ones. The sample presented here is the mixture of authentication data drawn from two Gaussian components: one representing genuine, and another fraudulent user attributes. Simulations have been performed using 1 and 3 dimensional data sets. The attributes used to describe 3-D data samples are:

(i) Number of incorrectly answered questions during the authentication session

(ii) Time duration of the authentication session and

(iii) Time spent on the system following the successful authentication

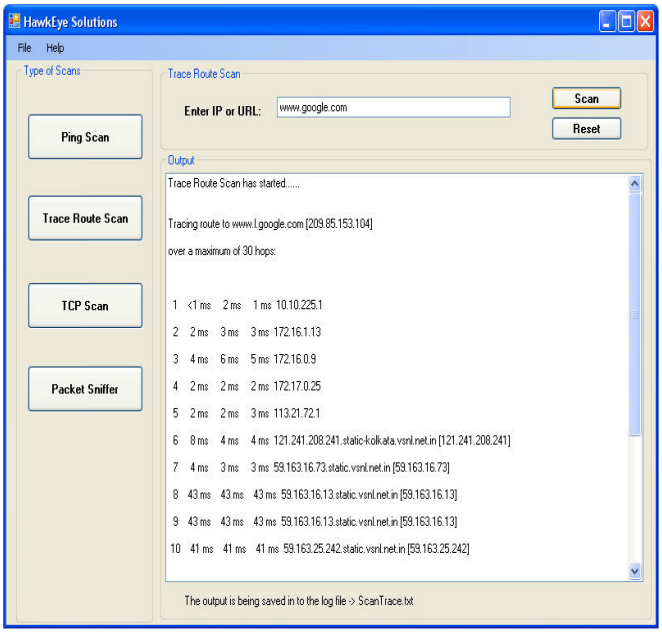

Figure 3 Screenshot of Trace Route Scan of HawkEye Solutions

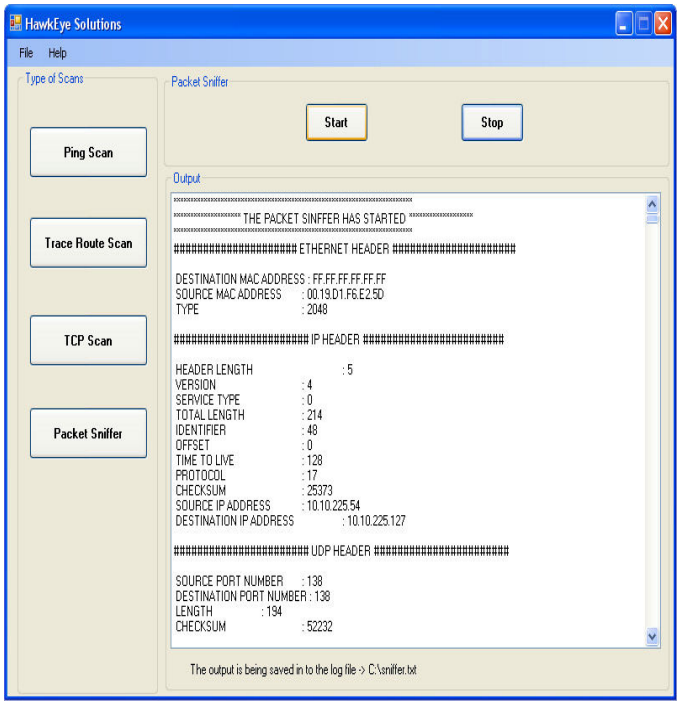

Figure 4 Screenshot of Packet Sniffing Utility of HawkEye Solutions

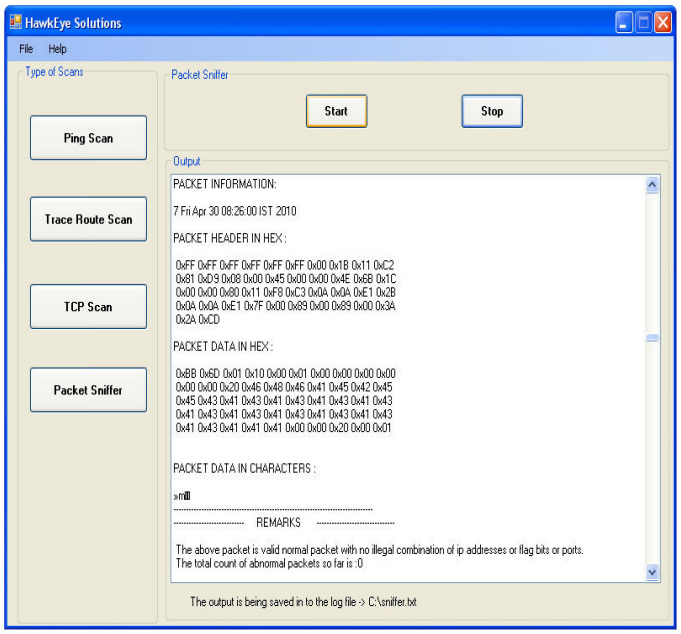

Figure 5 Screenshot of Abnormal Packet Detection by HawkEye Solutions

A single attribute – duration of the authentication session was used to describe 1-D data samples. Parameters for each attribute used in this example are given in Table 1.

To test the sensitivity of the EMID algorithm to the size of training data set, algorithm was initially trained using data sets of size $n_{training}$=50, 100, 150, 200 and 400 and then tested on the set of another $n_{testing}$=500 data samples. Total percentage of misclassified samples, i.e. false positives (FP) and false negatives (FN), is calculated for both training and testing phases and results averaged over 100 independent runs of the algorithm [11]. Tests have been performed for 1-D and 3-D Gaussian mixtures. Percentages of FPs and FNs for both cases (1-D and 3-D) are shown on figure 6.

Table 1: Parameters used in Simulation

Attribute Names	***Genuine*** ***(μG, σG)***	***Fraudulent*** ***(μF, σF)***
Attribute 1	(2, 1)	(5, 1.67)
Attribute 2	(12, 3.4)	(18,2.767)
Attribute 3	(3, 0.73)	(5, 3.4)

The most significant parameter for the intrusion detection application of EM algorithm is the number of attributes used to describe each authentication attempt, i.e. dimensionality of mixture data sets. Percentage of misclassified authentication attempts is significantly reduced when three attributes (3-D) are used compared to single attribute case (1-D). Another important factor is the size of training data set. Larger number of samples used in training data set results in reduced misclassification error, although increasing the size of data set above 100 does not improve the performance of the algorithm significantly, especially for 3-D Gaussian mixtures. Shown in fig. 7

Fig. 7 show the performance of EM based IDS for different ratios of genuine and fraudulent data sessions in the training and test sets. This parameter does not have a significant influence on the system performance, although some improvement is evident when the percentage of fraudulent authentication attempts in mixtures is reduced. Fig. 8 shows the convergence phenomenon achieved in eighteen iterations.

Table 2 presents the comparative analysis chart vis-à-vis design parameters of HawkEye Solutions with various other IDS/IPS.

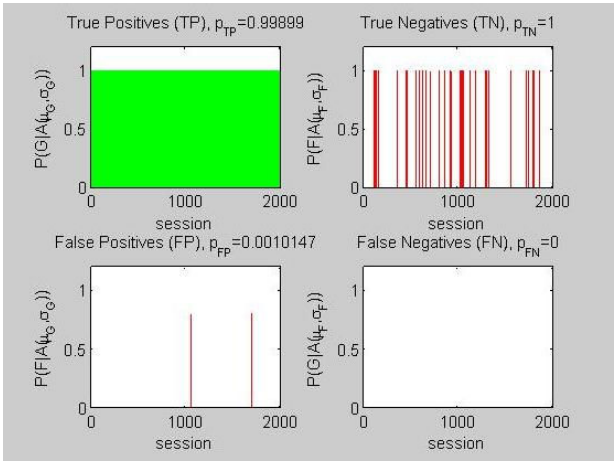

Figure 6 Mixture of FP & FN

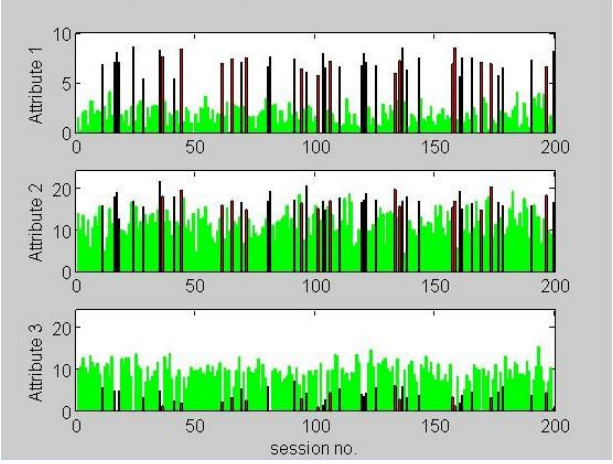

Figure 7 Performance of EM for different ratios of genuine and fradulent data sessions

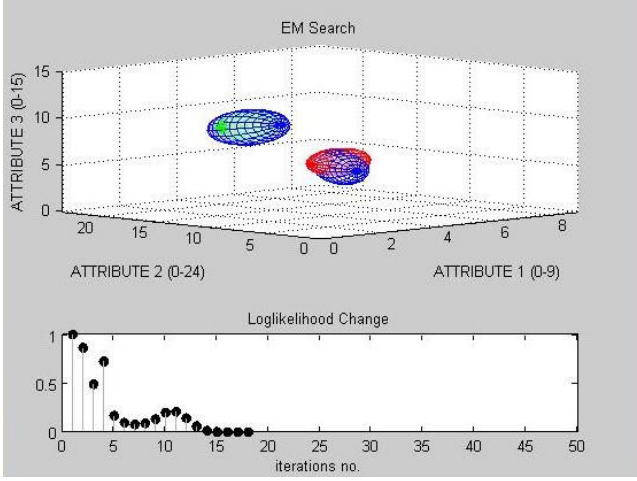

Figure 8 Convergence achieved in 18 iterations

Table 2: Performance Analysis of HawkEye Solutions vis-à-vis Design Parameters

Design Parameters	Performance Analysis of various IDS/IPS				
	Snort Inline (IDS)	Strata Guard (IDS)	IntruPro (IPS)	HawkEye Solutions (IDS)	Packet Alarm (IDS)
Anomalies Detection.	√	√	√	√	√
Firewall Inclusion					
IP Tunnels Inspection					
IPv6 Support		√			
Protection against DoS Attack			√	√	√
Personalized Rule Creation	√			√	
Automatic Rules Actualization	√		√	√	
Vulnerabilities Scanner				√	√
Multi-sensor Management		√	√		√
Secure Management		√			√
Remote Management		√	√	√	√
Reports Generation		√	√	√	

6. CONCLUSION

An NIDS always has a future scope of preventing a network intrusion from happening which falls under the heading of an Intrusion Prevention System (IPS). The utilities provided in the software are merely tools to detect suspicious activities on the network, but if it succeeds in detecting any abnormal behavior, the job that still remains is to handle the intrusion and thus avoiding any possible damage to the resources which would have occurred if the intruder succeeded.

The primary step to be taken after detecting an intrusion is to find means to negate the slightest of chances that the intruder may get to exercise his evil intensions. As far as IDS is concerned, the sniffing of packets can be utilized by the network administrator to analyze the network behavior. Also, a bench mark can be set for normal packet activities on various ports and it can be stored in a database which can be accessed and compared with real time traffic on the network to measure any abnormal deviations which occur during various FLOOD ATTACKS.

Early detection is the key in such situations as FLOOD ATTACKS if detected early can be taken care of thus avoiding the possibility of such attacks using up the resources of the host to an extent that it virtually becomes in-operative.

Anyone who is involved with intrusion detection should have a solid knowledge of what normal and abnormal IP traffic is. It is not uncommon for the administrator of an intrusion detection system to get great alerts from the IDS but not understand what they really mean. Although there are plenty of tools available in the market that promises to provide great alerts but applications centered towards increasing the Administrators' capabilities is difficult to find.

EMID, the novel IDS algorithm proposed by the authors proved that it is undeniably a method for intrusion detection which identifies fraudulent authentication attempts. The Matlab implementation of EMID was based on collected data from MIT [12]. The next challenge for this research is to prove this assumption by collecting real data in an unbiased manner. Alternatively, algorithm might have to be modified to account

for differently shaped user distributions. The proposed methods suggested here states that there is potential value in applying quantitative techniques to the development of systems that are more robust and resilient than traditional password authentication. This can also be used in developing Intrusion Detection & Prevention System which is the next challenge for the authors.

Easy accessibility condition in this kind of networks causes their vulnerability against wired networks to high. The level of vulnerability has made it necessary to consider security issues in wireless networks more than before and nowadays new group added to NIDS categories for Wireless networks that named WIDS (Wireless Intrusion Detection System) [13]. WIDS which monitors wireless network traffics and analyze them to identify suspicious activity involving the wireless networking protocols. The ultimate goal of the authors is to design HawkEye Solutions as an Intrusion Detection and Prevention System (IDPS) by integrating EMID with the former.

7. ACKNOWLEDGMENTS

Our sincere thanks to Mr. Maninder Singh Auluck and Arvind R who have contributed towards development of HawkEye Solutions.

8. REFERENCES

[1] Tim Crothers, "Implementing Intrusion Detection Systems", ISBN: 0-7645-4949-9, John Wiley & Sons, Inc. 2003.

[2] Stephen Northcutt & Judy Novak, "Network Intrusion Detection", 3/e, ISBN: 9780735712652, Sams Publishing 2005.

[3] Wikepedia Expectation-maximization algorithm, http://en.wikiped ia.org/wiki/Expectation-maximization_algorithm.

[4] Robin Hanson, Sterling Software, John Stutz "Bayesian Classification Theory" Technical Report FIA-90-12-7-01. Artificial Intelligence Research Branch NASA Ames Research Center, Mail Stop 244-17, Moffet Field, CA 94035, USA

[5] Pantos, "Packet Reading with libpcap", April 2010 http://www.systhread.net/texts/200805lpcap1.php

[6] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2 ed: John Wiley & Sons, 2001

[7] M. Aitkin and D. B. Rubin, "Estimation and hypothesis testing in finite mixture models," *Journal of the Royal Statistical Society*, vol. 47, pp. 67-75, 1985.

[8] M. A. T. Figueiredo and A. K. Jain, "Unsupervised Learning of Finite Mixture Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 381-396, 2002.

[9] S. Theodoridis and K. Koutrombas, Pattern Recogni-tion, 3 ed: Elsevier Academic Press, 2006.

[10] RFC 768, "User Datagram Protocol", http://www.faqs.org/rfcs/ rfc768.html.

[11] Edward Guillen, Daniel Padilla, Yudy Colorado, "Weakness and Strength Analysis over Network-based Intrusion Detection and Prevention System", 2009, IEEE Latin-American Conference on Communications, 2009. LATINCOM '09, ISBN 978-1-4244-4387-1.

[12] 1998 DARPA Intrusion Detection Data Sets, " http://www.ll.mit. edu/mission/communications/ist/corpora/ideval/data/index. html.

[13] Yu-Xi Lim, Tim Schmoyer, John Levine, Henry L. Owen. "Wireless Intrusion Detection and Response", Proceedings of the 2003 IEEE Workshop on Information Assurance.