International Journal of Engineering Sciences & Emerging Technologies, April 2012.
ISSN: 2231 – 6604 Volume 2, Issue 1, pp: 29-36 ©IJESET

ANOMALY DETECTION USING ARTIFICIAL NEURAL NETWORK

Manoranjan Pradhan[1], Sateesh Kumar Pradhan[2], Sudhir Kumar Sahu[3]
[1]Department of Comp. Sc. & Engg., GITA, Bhubaneswar, India
manoranjanpradhan72@gmail.com
[2]Department of Computer Science, Utkal University, India
sateeshind@yahoo.com
[3]P.G. Department of Statistics, Sambalpur University, India
drsudhir1972@gmail.com

ABSTRACT

In this research, anomaly detection using neural network is introduced. This research aims to experiment with user behaviour as parameters in anomaly intrusion detection using a backpropagation neural network. Here we wanted to see if a neural network is able to classify normal traffic correctly, and detect known and unknown attacks without using a huge amount of training data. For the training and testing of the neural network, we used the DARPA Intrusion Detection Evaluation data sets. In our final experiment, we have got a classification rate of 88% on known and unknown attacks. Compared with other researches our result is very promising.

KEYWORDS: *Network Security, Intrusion Detection System, Artificial Neural Networks, Backpropagation Neural Network, Anomaly Detection, Datasets, Detection Rate.*

I. INTRODUCTION

Organizations usually wish to preserve the confidentiality of their data which is very vital to an organization. With the widespread use of the internet, it has become a key challenge to maintain the secrecy and integrity of organizations' vital data. Network security has been an issue almost since computers have been networked together. Since the evolution of the internet, there has been an increasing need for security systems. Conventional techniques for network security include security mechanisms like user authentication, cryptography and intrusion prevention systems like firewalls. This paper presents the scope and status of our research in anomaly detection. Here, Anomaly detection uses models of the intended behaviour of users and applications, interpreting deviations from the "normal" behaviour as a problem [1]. Maxion and Tan [2] have expanded this definition: "An anomaly is an event (or object) that differs from some standard or reference event, in excess of some threshold, in accordance with some similarity or distance metric on the event". The task of anomaly intrusion detection is to determine if an activity is unusual enough to suspect an intrusion. A basic assumption of anomaly detection is that attacks differ from normal behaviour [3]. If an organization implements an anomaly based Intrusion Detection System, they must first build profiles of normal user and system behaviour to serve as the statistical base for intrusion detection, and then use deviations from this baseline to detect possible intrusions[4]. Any activity sufficiently deviant from the baseline will be reported as anomalous and considered as a possible attack. The trade-off between the ability to detect new attacks and the ability to generate a low rate of false alarms is the key point to develop an anomaly Intrusion detection system.
This paper describes Anomaly Detection Using Artificial Neural Network. Here we wanted to see if a neural network was able to classify normal traffic correctly, and detect known and unknown attacks

without using a huge amount of training data. System performance has been tested and satisfactory results have been obtained. Section 2 describes IDS in general. Section 3 describes ANN in general. Section 4 presents an overview of frequently occurring network attacks. Section 5 describes our proposed method. Section 6 presents the experiment and results. Section 7 presents comparison with other research. Section 8 provides the concluding remarks and section 9 presents limitations and further work.

II. INTRUSION DETECTION SYSTEM

Intrusion detection system (IDS) can be software or hardware that monitors for intrusions and anomalies from the environment it is set to guard. In general the IDS is a security monitoring tool like a firewall that tries to detect and possibly prevent malicious activity. Two main techniques for intrusion detection exist based on what they can detect. These two techniques are misuse detection and anomaly detection. Misuse detection and anomaly detection systems can be further divided into two groups based on the detection method; into behaviour based and into knowledge based IDS. Behaviour based IDS monitor behaviour deviations of the system in order to detect intrusions and anomalies. Knowledge based IDS monitors a system using patterns of known intrusions[5]. Basic functionality of IDS is to act as a passive alerting system. This means that once intrusion is detected the IDS generates an alarm and provides all the relevant information (time, IP packets, etc.) that triggered the alarm. IDS that operates in active mode, reacts to detected intrusions by using countermeasures to prevent the access of the intrusive data accessing the system. Active IDSes are called intrusion prevention systems (IPS). For example, IPS can alter the firewall rules, change routing tables, limit network bandwidth or just disconnect the connection. IDSes can be further divided into two systems depending on where the IDS is placed. The IDS can be either a Network based IDS (NIDS) or Host based IDS (HIDS). Network intrusion detection system monitors for intrusions in network traffic and host intrusion detection system monitors the behaviour of a local machine [5]. When referring to the performance of IDSs, the following terms are often used when discussing their capabilities:

- True positive (TP): classifying an intrusion as an intrusion. The true positive rate is synonymous with *detection rate*, *sensitivity* and *recall*, which are other terms often used in the literature.
- False positive (FP): incorrectly classifying normal data as an intrusion. Also known as a *false alarm*.
- True negative (TN): correctly classifying normal data as normal. The true negative rate is also referred to as *specificity*.
- False negative (FN): incorrectly classifying an intrusion as normal.
- Accuracy=$\frac{TP+TN}{TP+FP+TN+FN} = \frac{\text{No. of correct classifications}}{\text{No. of all instances}}$

  Accuracy is also referred to as an *overall* classification rate.

III. ARTIFICIAL NEURAL NETWORK

One of the most sustained and ambitious inquiries of humankind has been the secrets of its brain, its learning and thought processes. Artificial Neural Networks (ANN) has been evolved in view of achieving human like performance in certain tasks where they outperform the conventional computers. The progress in the area of ANN is attributed to long efforts made by neurobiologists as well as neuroanatomists in developing models of human learning. McCulloch and Pitts (1943) conceived the artificial neuron derived as prototype to the biological neurons. Here we use the Backpropagation Neural Network for training and testing of the DARPA datasets.

3.1. Backpropagation Neural Network

Most popular training method for neural networks, the generalized delta rule [Rum86], also known as Backpropagation algorithm which is explained here briefly for feed-forward Neural Network (NN). The explanation here is intended to give an outline of the process involved in Backpropagation algorithm. The (NN) explained here contains three layers. These are input, hidden, and output layer.

During the training phase, the training data is fed into to the input layer. The data is propagated to the hidden layer and then to the output layer. This is called the forward pass of the Backpropagation algorithm. In forward pass, each node in hidden layer gets input from all the nodes from input layer, which are multiplied with appropriate weights and then summed. The output of the hidden node is the nonlinear transformation of this resulting sum. Similarly each node in output layer gets input from all the nodes of the hidden layer, which are multiplied with appropriate weights and then summed. The output of this node is the non-linear transformation of the resulting sum. The output values of the output layer are compared with the target output values. The target output values are used to teach network. The error between actual output values and target output values is calculated and propagated back toward hidden layer. This is called the backward pass of the Backpropagation algorithm. The error is used to update the connection strengths between nodes, i.e. weight matrices between input-hidden layers and hidden-output layers are updated. During the testing phase, no learning takes place i.e., weight matrices are not changed. Each test vector is fed into the input layer. The feedforward of the testing data is similar to the feed-forward of the training data. Backpropagation architecture was developed in the early 1970s by several independent sources (Werbor; Parker; Rumelhart, Hinton and Williams). There are many laws (algorithms) used to implement the adaptive feedback required to adjust the weights during training. The most common technique is backward-error propagation, more commonly known as back propagation. The Backpropagation algorithm searches for weight values that minimize the total error of the network over the set of training examples (training set). Backpropagation consists of the repeated application of the following two passes:
- Forward pass: in this step the network is activated on one example and the error of (each neuron of) the output layer is computed.
- Backward pass: in this step the network error is used for updating the weights (credit assignment problem).
Therefore, starting at the output layer, the error is propagated backwards through the network, layer by layer. This is done by recursively computing the local gradient of each neuron. Backpropagation algorithm uses supervised training, if the output is not correct, the weight are adjusted according to the formula:

$$W\mathbf{new} = W\mathbf{old} + \alpha\ (\text{desired} - \text{output}) * \text{input}.$$

This process is repeated for a number of iterations until the error is minimum admissible for all the patterns.

IV. DATASET

The Information System Technology Group at Massachusetts Institute of Technology – Lincoln Laboratory, sponsored by Defence Advanced Research Project Agency (DARPA) and Air Force Research Laboratory, has collected and evaluated the first standard corpora for evaluation of computer network Intrusion Detection Systems. This is called the DARPA Intrusion Detection Evaluation[6]. The data sets used in this research are the data sets from the 1998 DARPA Intrusion Detection Evaluation Program. In the dataset, the following attacks are present according to the actions and goals of the attacker. Each attack type falls into one of the following four main categories[7] :

4.1. Probing

Probing is a class of attacks where an attacker scans a network to gather information or find known vulnerabilities. An attacker with a map of machines and services that are available on a network can use the information to look for exploits. There are different types of probes: some of them abuse the computer's legitimate features, some of them use social engineering techniques. This class of attacks is the most commonly heard and requires very little technical expertise. Attacks used was IP sweep, Mscan, Nmap, Saint and Satan.

4.2. Denial of Service Attacks

Denial of Service (DoS) is a class of attacks where an attacker makes some computing or memory resource too busy or too full to handle legitimate requests, thus denying legitimate users access to a machine [8]. There are different ways to launch DoS attacks:

- Abusing the computers legitimate features.
- Targeting the implementations bugs.

- Exploiting the system's misconfigurations.

DoS attacks are classified based on the services that an attacker renders unavailable to legitimate users. An attack used was Apache2, Back, Mail bomb, Neptune, Ping of death, Process table, Smurf, Syslogd and UDP storm.

4.3. User to Root Attacks

User to root exploits are a class of attacks where an attacker starts out with access to a normal user account on the system and is able to exploit vulnerability to gain root access to the system. Most common exploits in this class of attacks are regular buffer overflows, which are caused by regular programming mistakes and environment assumptions. Attacks used was Perl and Xterm.

4.4. Remote to User Attacks

A remote to user (R2L) attack is a class of attacks where an attacker sends packets to a machine over a network, then exploits machine's vulnerability to illegally gain local access as a user. There are different types of R2L attacks; the most common attack in this class is done using social engineering. Attacks used was Dictionary, FTP-write, Guest, Imap, Named, Phf, Sendmail, Xlock and Xnsnoop.

V. THE PROPOSED METHOD

In order to build anomaly detection system using neural network, certain steps should be taken (Figure 1). First step, We randomly collected sessions from the 1998 DARPA Intrusion Detection Evaluation dataset[9], to train and test neural network. Second step is preprocessing the collected data, which is in binary tcpdump format, to a neural network readable format. Third step is determining the neural network structure, which is actually determining the number of hidden layers, number of hidden nodes in each layer, activation functions used in neural network and training algorithm. Fourth step is training neural network until a certain number of iterations or a certain RMSE value reached. Fifth and the final step is testing the neural network. The testing was conducted in three parts. In the preliminary experiment we just wanted to see when the neural network was properly trained to detect attacks and when it did not detect any attacks. The next experiment was done with a small amount of traffic, and in the end we conducted the final experiment where we used a higher amount of traffic. In these last two experiments we had normal traffic, known attacks and unknown attacks in three different files.

The backpropagation neural network is used to accomplish this task. Backpropagation is a supervised learning method i.e. training the neural net with help of teacher.

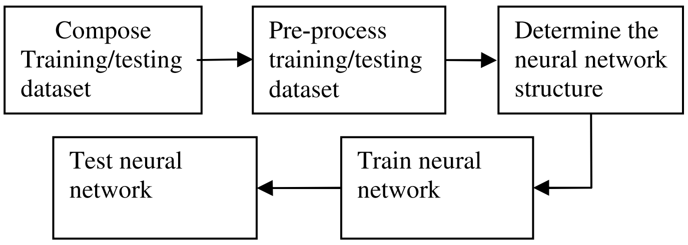
Compose Training/testing dataset
Pre-process training/testing dataset
Determine the neural network structure
Train neural network
Test neural network

Figure 1. Steps to be taken to build neural network based anomaly detection System

There are 7 weeks of traffic logs available from the 1998 DARPA Intrusion Detection Evaluation Data Sets [9]. For the experiments conducted in this research, only data sets from week 3 were collected. From these log files we used 12 different attacks. Some of these attacks was used both for training of the neural network and for testing the neural network with known attacks. The rest of the attacks were used for the testing of the neural network with unknown attacks. Because the DARPA log files had more information than we needed, we had to "clean" the log files. This was done by writing a Java application that could delete the parameters we did not want from the log file. When the DARPA log files were "cleaned" and just containing the parameters we wanted, they were ready to be converted into binary format. For the training of the neural network, we collected 206 sessions from the data sets.

Here MATLAB 7.6 Neural Network Toolbox was used for the implementation of the backpropagation neural networks. Using this tool one can define specifications like number of layers, number of neurons in each layer, activation functions of neurons in different layers, and number of training epochs. Then the training feature vectors and the corresponding desired outputs can be fed to the neural network to begin training. The first part of the training pair was a 121 bits or units long representation of one session in the traffic log from DARPA Intrusion Detection Evaluation data sets which is input layer for our neural network. The second part of the training pair was a 2 bit or units long output vector. This part tells the neural network if the representation of the session is an attack or normal traffic. If it is classified as normal traffic the second part of the training pair would be 0 1, if it is an attack it would be 1 0 which is output layer of our network. Number of inputs tells how many bits there are in the first part of the training pair. Number of outputs tells how many bits there are in the second part of the training pair. Number of hidden units is used to test how well trained the neural network can be. We used different numbers here in the testing. Number of training pairs tells how many sessions we used in the training of the neural network. We used 206, which was 104 sessions with normal traffic and 102 sessions with attacks. RMS-error is a value that can be adjusted to how we want to train the neural network. Number of iterations tells how many times the neural network should run to learn the input.
When using a backpropagation neural network, the usual criteria for termination of the training is that the RMS error is reduced to an acceptable level. There is no standard for the RMS error, but usually the lower it is, the better the classification rate is. But a too low RMS error could also over train the neural network causing the network to detect things that are exactly identical to the training data. In this work 0.0001RMS error is considered as performance goal. Another criterion for training termination is the number of iterations. We choose to use a fixed number of iterations. When the fixed number of iterations was reached, the training was stopped.

VI. EXPERIMENT AND RESULTS

We used 8 different files for the experiment: training file, preliminary experiment file, 3 files for the second experiment and 3 files for the final experiment. The training file had all the training data (normal traffic and attacks). The preliminary experiment had 60 sessions, where 30 sessions were known attacks and 30 sessions were unknown attacks. In the second experiment we used three different files. One file with normal traffic, one file with known attacks and one file with unknown attacks. This experiment had 20 sessions with normal traffic, 10 sessions with known attacks and 10 sessions with unknown attacks. In the final experiment we also used three different files. The difference was that in the normal file we now had 50 sessions with traffic, in the known attack file we had 25 sessions and in the unknown attacks file we also had 25 sessions.
The experiments were conducted in three parts. The preliminary experiment was conducted to see how many iterations and how many hidden units that was needed before the neural network was properly trained. The second and the final experiments were conducted to see how many percent of the normal traffic and the attacks that were classified correctly.

Table 1. The preliminary experiment results

Hidden Units	**Iterations**	**RMS-error**	**Classification rate for Training Set**	**Classification rate (%) for Training Set**
2	100	0.343452731524	0/60	0
3	100	0.343167226656	0/60	0
4	100	0.020895822140	53/60	88
5	100	0.018439157954	53/60	88
6	100	0.017222100971	53/60	88

When we used just three hidden units, no attacks was detected at all. With 4 hidden units, we got a detection rate on 88%. The results from this experiment gave the background for choosing the number of hidden units and iterations used for the training of the neural network in the last two experiments. This meant that number of hidden units had to be over 4, and number of iterations had to be over 100. There is a huge drop in the RMS-error between 3 and 4 hidden units, and this shows that a lower RMS-error affects the detection rate.

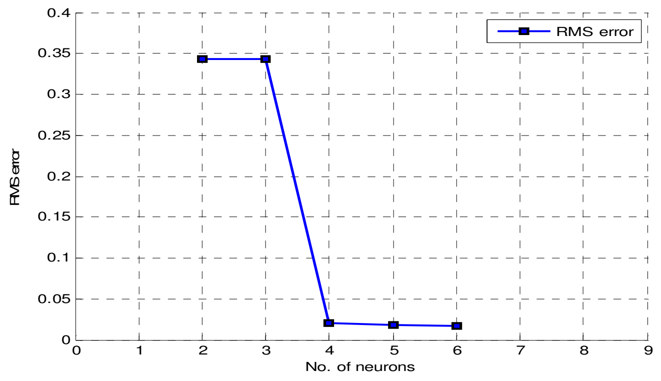

Figure 2. RMS-error for the preliminary experiment

This was just a brief testing where we did several tests and for each test changed the number of iterations for the neural network with 100, and number of hidden units with one. The results here would probably be different if we tried to increase/decrease the number of iterations with just one, and test all iterations with different number of hidden units. But this was just done to see differences in the RMS-error rate when attacks were not detected and when attacks were detected.

In the second experiment, The system has a classification rate of 100% for normal traffic, 90% for known attacks and 60% for unknown attacks and in the final experiment , The system has a classification rate of 100% for normal traffic, 96% for known attacks and 80% for unknown attacks. So our analysis of the results of the experiments are shown and also which attacks are not detected by the neural network. As explained before, the lower the RMS rate is, the better the detection rate normally is.

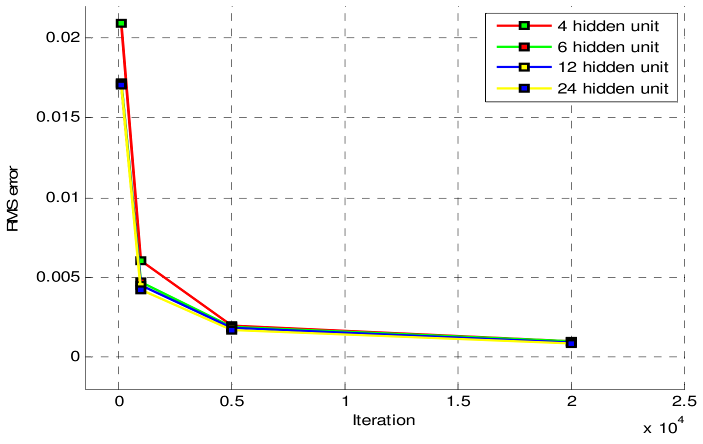

Figure 3. RMS-error comparison

The RMS-error dropped pretty much when we increased the hidden units from 4 to 6, but there were small changes in the RMS-error between using 6 hidden units and using 24 hidden units.

In our experiments, we classified all normal traffic correctly. Both in the second experiment and the final experiment we had a classification rate of 100 %, which gives a false positive rate of 0 %.

In the final experiment, One session of known attack that was not detected, an attack method called warezmaster. Warezmaster is anonymous upload of warez (usually illegal copies of copywrited software) onto a FTP server[9].

The first unknown attack missed was an ipsweep. The second missed attack was a Denial of Service attack called smurf. The last three missed attacks were all Denial of Service attack called neptune.

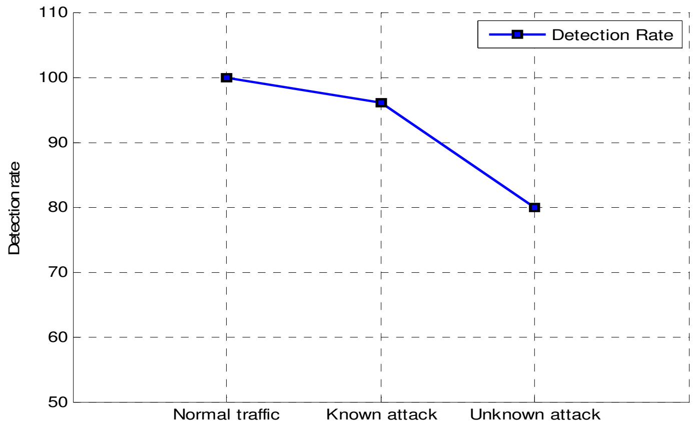

Figure 4. Detection rate on dataset for anomaly detection system

VII. COMPARISON WITH OTHER RESEARCH

There has been done a lot of research on intrusion detection. Ghosh and Schwartzbard used system behaviour as input for the neural network in their experiment [10]. They used 4 weeks of training data and 161 sessions with testing data. Out of these, they only had 22 attack sessions. The results from this experiment showed a detection rate of 77.3% and a false positive rate of 3.6 %. In their experiments, they used software behaviour for the input to the neural network, not user behaviour as we used in our experiments.

Lippmann and Cunningham have done some experiments using keyword selection and neural networks [11]. The results from their experiments showed a detection rate of 80%, but they had roughly 10 false alarms per day. Unfortunately they did not calculate the false positive rate into percentage, so this could not be compared with our results and the results from Gosh and Schwartzbards experiments.

In our research we had a detection rate of 88% on 50 attacks (both known and unknown attacks), but we had a false positive rate of 0 %. This means that all normal sessions in our experiment were classified correctly as normal traffic. Compared with two other researches where they got classification rates of 77.3% and 80%, the results we got in our experiments are very promising.

VIII. CONCLUSION

As showed in this research, neural networks can successfully be used as a method for training and learning an Intrusion Detection System. The main problem with today's Intrusion Detection System is that they produce many false alarms, and this takes up much of a system administrator's time and resources.

Here the neural network had a classification rate of 100 %, which gives a false positive rate of 0 %. This means that none of the normal sessions were classified as an attack. If normal traffic was classified as an attack a false alarm would be raised.

In this research, we have tested the ability of a backpropagation neural network to classify normal traffic correctly and to detect attacks without a huge amount of training data. The results of our study show that a neural network do not need huge amount of training data to be able to classify traffic correctly. Completely unknown attack has been detected, among them Denial of Service attacks. Still one Denial of Service attack type called neptune went by undetected, and this shows that more testing is needed to find out why this happened.

IX. LIMITATIONS AND FURTHER WORK

In this work there is need to regular update of the signature database, need to consider known and unknown attack. Only detect, cannot prevent the intrusions and it's an offline system. In the future work this system can be extended to an online system by little effort. There has been a lot of research on intrusion detection, and also on the use of neural networks in intrusion detection. As showed in this thesis, backpropagation neural networks can be used successfully to detect attacks on a network. The same experiments should also be conducted with other types of neural networks to see if these types can improve the detection rate we got from the experiments with a backpropagation neural network.

REFERENCES

[1] R. A. Kemmerer and G. Vigna, "Intrusion Detection: A Brief Introduction and History", Security & Privacy – Supplement – IEEE Computer Magazine, pp. 27-30, 2002.
[2] C. A. Carver, J. M. D. Hill and U. W. Pooch, "Limiting Uncertainty in Intrusion Response", 2001 IEEE Man Systems and Cybernetics Information Assurance Workshop, pp. 142-147, New York, June 2001.
[3] E. Biermann, E. Cloete and L. M. Venter, "A Comparison of Intrusion Detection Systems", Computers & Security, Vol. 20, pp. 676-683, 2001.
[4] C. Herringshaw, "Detecting Attacks on Networks", IEEE Computer, Vol 30, No 12, pp. 16-17, 1997.
[5] A. Sundaram, "An introduction to intrusion detection, Crossroads", Volume.2, Issue 4, pp. 3-7, April 1996.
[6] H. Debar, M. Becker and D. Siboni, "A Neural Network Component for an Intrusion Detection System", Proc. 1992 IEEE Computer Society Symposium on Research in Computer Security and Privacy, pp. 240-250, Oakland, May 1992.
[7] R. P. Lippmann and R. K. Cunningham, "Improving Intrusion Detection Performance using Keyword Selection and Neural Networks", Computer Networks, Vol. 34, No. 4, pp. 597-603, October 2000.
[8] Sandeep Sharma, "Application of Neural Networks to Intrusion Detection Classification and detection of computer intrusions", 2005.
[9] H. Debar, M. Becker and D. Siboni, "A Neural Network Component for an Intrusion Detection System", Proc. 1992 IEEE Computer Society Symposium on Research in Computer Security and Privacy, pp. 240-250, Oakland, May 1992.
[10] A. K. Ghosh and A. Schwartzbard, "A Study in using Neural Networks for Anomaly and Misuse Detection", Proc. 8th USENIX Security Symposium, Washington, USA, 23-26 August 1999.
[11] R. F. Erbacher, K. L. Walker and D. A. Frincke, "Intrusion and Misuse Detection in Large-Scale Systems", IEEE Computer Graphics and Applications, Vol. 22, No. 1, pp.38-48, January/February2002.

Authors

Manoranjan Pradhan holds a M.Tech Degree in Computer Science. He is presently working as an Associate Professor & Head in the Department of Computer Science and Engineering, Gandhi Institute for Technological Advancement, Bhubaneswar, India. He has 13 years of teaching experience. Currently, he is pursuing Doctoral research in Computer Science. He has published many papers in national and international journals. His research interests include Computer Security, Intrusion Detection and Soft Computing.

Sateesh Kumar Pradhan obtained his Ph.D. Degree in Computer Science from Berhampur University, India during the year 1999. He joined Berhampur University, as Lecturer in the year 1987 and promoted to Reader in 1999. He was Head, Post Graduate Department of Computer Science, Utkal University, India during 2001-2003. He was the Organizing Chair of the International Conference on Information Technology-2005. He served as a senior faculty in the Computer Engineering Department of King Khalid University, Ministry of Higher Education, Saudi Arab from September 2006 to July 2011. At present he is the Head, Post Graduate Department of Computer Science, Utkal University, Bhubaneswar, India. His research interest includes Neural Computing, Computer Architecture, Ad hoc Networks, Computer Forensic.

Sudhir Kumar Sahu is a Lecturer in the Department of Statistics, Sambalpur University, Orissa, India. He did his M.Sc in Statistics, M,Tech in Computer Science and Ph.D in Operational Research from Utkal University, Orissa. He has so far guided 10 Ph.D theses in the areas of Computer Science and Inventory Control. He has published several papers on inventory modeling in national and international journals of repute.