ISSN 2047-3338

Layered Secure Pattern for Amending Intrusion Detection

Subha Sree Mallela[1] and Sravan Kumar Jonnalagadda[2]

[1]Department of CSE, DMSSVH College of Engineering, Machilipatnam, AP., India
[2]Department of IT, DMSSVH College of Engineering, Machilipatnam, AP., India

***Abstract*– The growth of data communications has involved an increase in unauthorized accesses and data manipulation with the resulting security violations. Many methods have been developed to secure the network infrastructure and communication over the Internet. Since it is impossible to predict and identify all the vulnerabilities of a network, and penetration into a system by malicious intruders cannot always be prevented, intrusion detection systems (IDSs) are essential entities for ensuring the security of a networked system. An IDS is software (or hardware) designed to detect unwanted attempts at accessing, manipulating, or disabling of computer systems, mainly through a network. However, an accurate system that cannot handle large amount of network traffic and is slow in decision making will not fulfil the purpose of an intrusion detection system. In this paper, we present a layered framework for building intrusion detection systems which can detect a wide variety of attacks reliably and efficiently when compared to the traditional network intrusion detection systems. Another advantage of this Layer based Intrusion Detection System (LIDS) framework is that it is very general and easily customizable depending upon the specific requirements of individual networks. We, thus, address these shortcomings and develop better intrusion detection systems which are accurate in attack detection, efficient in operation and have wide attack detection coverage.**

***Index Terms*– Intrusion Detection, Hybrid System, Anomaly, Layered Approach, Probe Attack and Logging System**

I. INTRODUCTION

INTRUSION Detection System is software that detects an attack on a network or computer system. Intrusion detection (ID) is a type of security management system for computers and networks. An ID system gathers and analyses information from various areas within a computer or a network to identify possible security breaches, which include both intrusions (attacks from outside the organization) and misuse (attacks from within the organization). ID uses vulnerability assessment (sometimes referred to as scanning), which is a technology developed to assess the security of a computer system or network. Intrusion detection functions include: Monitoring and analysing both user and system activities, analysing system configurations and vulnerabilities, assessing system and file integrity, analysis of abnormal activity patterns and tracking user policy violations.

Intrusion detection systems are classified as network based, host based, or application based depending on their mode of deployment and data used for analysis [2]. Additionally, intrusion detection systems can also be classified as signature based or anomaly based depending upon the attack detection method. The signature-based systems are trained by extracting specific patterns (or signatures) from previously known attacks while the anomaly-based systems learn from the normal data collected when there is no anomalous activity [2].

Another approach for detecting intrusions is to consider both the normal and the known anomalous patterns for training a system and then performing classification on the test data. Such a system incorporates the advantages of both the signature-based and the anomaly-based systems and is known as the Hybrid System [2]. Hybrid systems can be very efficient, subject to the classification method used, and can also be used to label unseen or new instances as they assign one of the known classes to every test instance. This is possible because during training the system, it learns features from all the classes. The only concern with the hybrid method is the availability of labelled data. However, data requirement is also a concern for the signature and the anomaly-based systems as they require completely anomalous and attack free data, respectively, which are not easy to ensure.

For an intrusion detection system, it is important to detect previously known attacks with high accuracy. However, detecting previously unseen attacks is equally important in order to minimize the losses as a result of a successful intrusion. In [1], a scenario is described in which a software agent can be used to attack a specific target without affecting any other network with a purpose to search and transmit confidential and sensitive information without authorized access. Such an attack can be carried out by experts with the motive to hide the entire attack and protect their identity from being discovered.

Further, since the attack targets only a single network, it would not be detected by large scale cooperative intrusion detection systems. The most significant part of the entire attack is that none of the present systems can detect such attacks and the agent can destroy itself when the attack is successful without leaving traces of its activities. Unlike worms, the replication in case of an intruding agent is limited and it does not degrade performance at the target making their detection very difficult. In order to operate in high speed networks, present anomaly based systems consider the events individually, thereby, discarding any correlation between the sequential events. In cases when the present systems consider a sequence of events, they monitor only one feature, ignoring others, which results in a poor model. Hence, in this paper efficient intrusion detection frameworks and methods are introduced which consider a sequence of events and which analyse multiple features without assuming any independence among the features.

A. *Concise Explanation of IDS*

Intrusion detection is the methodology by which undesirable or aberrant activity is detected on a host or a network. It can be defined as the tools, methods, and resources to help identify, assess, and report unauthorized or unapproved network activity. Intrusion detection is typically one part of an overall protection system that is installed around a system or device. The main use of intrusion detection systems (IDS) is to detect attacks against information systems and networks. Normal use of the network and its functioning can also be monitored with IDS.

IDSs collect network traffic information from some point on the network or computer system and then use this information to secure the network.

B. *IDS Requirements*

At least one past effort has identified desirable characteristics for IDS. Regardless of what mechanisms an IDS is based, it must do the following:

- Run continuously without human supervision,
- Be fault tolerant and survivable,
- Resist subversion,
- Impose minimal overhead,
- Observe deviations from normal behaviour
- Be easily tailored to a specific network
- Adapt to changes over time, and
- Be difficult to fool.

We have developed a similar set of requirements along two themes: functional and performance requirements

C. *Functional Requirements*

As the network-computing environment increases in complexity, so do the functional requirements of IDSs. Common functional requirements of an IDS being deployed in current or near-term operational computing environments include the following:

- The IDS must continuously monitor and report intrusions.
- The IDS must supply enough information to repair the system, determine the extent of damage, and establish responsibility for the intrusion.
- The IDS should be modular and configurable as each host and network segment will require their own tests and these tests will need to be continuously upgraded and eventually replaced with new tests.
- The IDS should be adaptive to network topology and configuration changes as computing elements are dynamically added and removed from the network.
- The IDS should be able to learn from past experiences and improve its detection capabilities over time. Self-tuning IDS will be able to learning from false alarms with the guidance of system administrators and eventually on its own.
- The IDS should be able to be easily and frequently updated with attack signatures as new security advisories and security patches become available and new vulnerabilities and attacks are discovered.
- Decision support tools will be necessary to help system administrators respond to various attacks. The IDS will be required not only to detect anomalous events, but also to take automated corrective action.
- The IDS should be able to perform data fusion and be able to process information from multiple and distributed data sources such as firewalls, routers, and switches. As real-time detection demands push networked-based solutions to re-programmable hardware devices that can download new capabilities, the IDS will need to be able to communicate with the hardware-based devices.
- Data reduction tools will be necessary to help the IDS process the information gathered from data fusion techniques. Data mining tools will be helpful in running statistical analysis tools on archived data in support of anomaly detection techniques.
- The IDS should be capable of providing an automated response to suspicious activity.
- Rapid changes in network conditions and limited network administration expertise make it difficult for system administrators to diagnose problems and take corrective action to minimize the damage that intruders can cause.
- The ability to detect and react to distributed and coordinated attacks will become necessary. Coordinated attacks against a network will be able to marshal greater forces and launch many more and varied attacks against a single target. These attacks can be permutations of known attacks, be rapidly evolving, and be launched at little cost to the attackers.
- Distributing the computational load and the diagnostic capabilities to agents scattered throughout the network adds a level of fault-tolerance, but it is often necessary for the system administrator to have control over the IDS from a central location.
- The IDS should be able to work with other Commercial Off-the-Shelf (COTS) security tools, as no vendor toolset is likely to excel in or to provide complete coverage of the detection, diagnosis, and response responsibilities. The IDS framework should be able to integrate various data reduction, forensic, host-based, and network-based security tools. Interoperability and conformance to standards will further increase the value of the IDS.
- IDS data often requires additional analysis to assess any damage to the network after an intrusion has been

detected. Although the anomalous event was the first detected, it may not be the first attempt to gain unauthorized access to the network. Post event analysis will be needed to identify compromised machines before the network can be restored to a safe condition.
- The IDS itself must also be designed with security in mind. For example, the IDS must be able to authenticate the administrator, audit administrator actions, mutually authenticate IDS devices, protect the IDS data, and not create additional vulnerabilities.

D. Performance Requirements

An IDS that is functionally correct, but that detects attacks too slowly is of little use. Thus we must enumerate several performance requirements for IDSs. The IDS performance requirements include:
- To the extent possible, anomalous events or breaches in security should be detected in real-time and reported immediately to minimize the damage to the network and the loss or corruption of confidential information.
- The IDS must not place undue burden or interfere with the normal operations for which the systems were bought and deployed to begin with. This requirement makes it necessary for the agents to be cognizant of the consumption of network resources for which they are competing.

The IDS must be scalable. As new computing devices are added to the network, the IDS must be able to handle the additional computational and communication load

II. Intrusion Detection Systems

Detecting intrusions in networks and applications has become one of the most critical tasks to prevent their misuse by attackers. The cost involved in protecting these valuable resources is often negligible when compared with the actual cost of a successful intrusion, which strengthens the need to develop more powerful intrusion detection systems. Intrusion detection started in 1980's and since then a number of approaches have been introduced to build intrusion detection systems [3]. However, intrusion detection is still at its infancy and naive attackers can launch powerful attacks which can bring down an entire network [2]. To identify the shortcoming of different approaches for intrusion detection, we explore the related research in intrusion detection. We describe the problem of intrusion detection in detail and analyse various well known methods for intrusion detection with respect to two critical requirements viz. accuracy of attack detection and efficiency of system operation. We observe that present methods for intrusion detection suffer from a number of drawbacks which significantly affect their attack detection capability.

In this section we provide a high-level categorization of IDSs and give an abstract idea of how they work. In the discussion we provide examples of existing IDSs.

Traditionally, there are two basic approaches to intrusion detection; *anomaly detection* and *misuse detection*.

In anomaly detection the goal is to define and characterize legitimate behaviours of the users, and then detect anomalous behaviours by quantifying deviations from the former. However, identifying the distance between anomalous and legitimate behaviours is a rather difficult notion to quantify.

Anomaly detection can be *static* or *dynamic*. A static anomaly detection system is based on the assumption that there is a static portion of the system being monitored. Static portions of the system can be represented as a binary string or a set of binary strings (like files). If the static portion of the system ever deviates from its original form, either an error has occurred or an intruder has altered the static portion of the system. Dynamic anomaly detectors are harder to build since building them requires a definition of behaviour, which is often defined as a sequence (or partially ordered sequence) of distinct events. Differentiating between normal and anomalous activity in dynamic anomaly detection systems is much harder than the problem of distinguishing changes in static elements. Dynamic anomaly detection systems usually create a *base profile* to characterize normal, acceptable behaviour. A profile usually consists of a set of observed measures of behaviour for a selected set of dimensions. After initializing the base profile the dynamic anomaly detection systems are similar to the static ones; they monitor the behaviour by comparing the current behaviour with that implied by the base profile. Typically, there is a wide variation of acceptable behaviours and statistical methods employed to measure deviation from the base profile. The main challenge in dynamic anomaly detection systems is that they must build accurate base profiles and then recognize behaviours that significantly deviate from the profile.

Misuse detection is concerned with identifying intruders who are attempting to break into a system using some known technique. If a system security administrator were aware of all the known vulnerabilities then a misuse detection system would be able to identify their occurrences and eliminate them. A fairly precisely known kind of intrusion is known as *intrusion scenario*. A misuse detection system compares current system activity to a set of intrusion scenarios in an attempt to identify a scenario in progress. The main advantage of a misuse detection system is that the system knows for a fact how normal behaviour should manifest itself. This leads to a simple and efficient processing of the audit data. The obvious disadvantage of such systems is that the specification of the signatures to be detected is a time-consuming task that requires lots of domain knowledge. At the same time, misuse detection systems lack the ability to identify novel intrusion profiles.

III. Related Research

Dorothy Denning proposed the concept of intrusion detection as a solution to the problem of providing a sense of security in computer systems. The basic idea is that intrusion behaviour involves abnormal usage of the system. Different techniques and approaches have been used in later developments. Some of the techniques used are statistical approaches, predictive pattern generation, expert systems, keystroke monitoring, state transition analysis, pattern matching, and data mining techniques. Fig. 1 illustrates a simple network, which is protected using IDS.

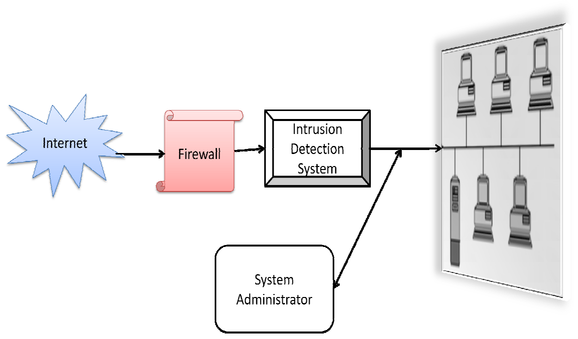

Fig. 1: Network protection using conventional IDS

Statistical approaches compare the recent behaviour of a user of a computer system with observed behaviour and any significant deviation is considered as intrusion [4]. This approach requires construction of a model for normal user behaviour. Any user behaviour that deviates significantly from this normal behaviour is flagged as an intrusion.

Predictive pattern generation uses a rule base of user profiles defined as statistically weighted event sequences [5]. This method of intrusion detection attempts to predict future events based on events that have already occurred.

State transition analysis approach uses the state transitions of the system to identify the intrusions. State transition diagrams list only the critical events that must occur for the successful completion of the intrusion [6].

Keystroke monitoring technique utilizes user's keystrokes to determine the intrusion attempt [7]. The main approach is to pattern match the sequence of keystrokes to some predefined sequences to detect the intrusion. The main problems with this approach are lack of support from operating system to capture the keystroke sequences and also many ways of expressing the sequence of keystrokes for same attack.

Expert systems have played an important role to build IDS [8]. The rules may recognize single auditable events that represent significant danger to the system by themselves, or they may recognize a sequence of events that represent an entire penetration scenario.

The Model-based approach proposed by Garvey and Lunt [9] attempts to model intrusions at a higher level of abstraction than audit trail records. The objective is to build scenario models that represent the characteristic behaviour of intrusions. This technique differs from the rule-based expert system technique, which simply attempt to pattern match audit records to expert rules.

The Pattern matching [10] approach encodes known intrusion signatures as patterns that are then matched against the audit data. Intrusion signature is represented with Petri Net, the start state and final state notion is used to define matching to detect the intrusion.

Data mining approaches for intrusion detection was first implemented in Mining Audit Data for Automated Models for Intrusion Detection [11]. Raw data is converted into ASCII network packet information, which in turn is converted into connection level information. These connection level records contain within connection features like service, duration etc. Data mining algorithms are applied to this data to create models to detect intrusions.

Neural networks have been used both in anomaly intrusion detection as well as in misuse intrusion detection. In the first approach of neural networks [12] for intrusion detection, the system learns to predict the next command based on a sequence of previous commands by a user.

Neuro-fuzzy computing is a popular framework for solving complex problems. An Adaptive neuro-fuzzy IDS is proposed in [13]. Multivariate Adaptive Regression Splines (MARS) [14] is an innovative approach that automates the building of accurate predictive models for continuous and binary dependent variables. It excels at finding optimal variable transformations and interactions, and the complex data structure that often hides in high-dimensional data.

Linear Genetic Programming (LGP) is a variant of the conventional Genetic Programming (GP) technique that acts on linear genomes. Its main characteristics in comparison to tree-based GP lies in fact that computer programs are evolved at the machine code level, using lower level representations for the individuals. This can tremendously hasten up the evolution process as, no matter how an individual is initially represented, finally it always has to be represented as a piece of machine code, as fitness evaluation requires physical execution of the individuals.

Naive Bayes classifiers have also been used for intrusion detection [15]. However, they make strict independence assumption between the features in an observation resulting in lower attack detection accuracy when the features are correlated, which is often the case for intrusion detection.

Bayesian network can also be used for intrusion detection [16]. However, they tend to be attack specific and build a decision network based on special characteristics of individual attacks. Thus, the size of a Bayesian network increases rapidly as the number of features and the type of attacks modelled by a Bayesian network increases. However, modelling the system calls alone may not always provide accurate classification as in such cases various connection level features are ignored.

Decision trees have also been used for intrusion detection [15]. The decision trees select the best features for each decision node during the construction of the tree based on some well-defined criteria. One such criterion is to use the information gain ratio, which is used in C4.5. Decision trees generally have very high speed of operation and high attack detection accuracy.

Other approaches for detecting intrusion include the use of genetic algorithm and autonomous and probabilistic agents for intrusion detection. These methods are generally aimed at developing a distributed intrusion detection system. To overcome the weakness of a single intrusion detection system, a number of frameworks have been proposed, which describe the collaborative use of network-based and host based systems [17].

IV. Layered Structure for Intrusion Detection

Layered Structure for building intrusion detection systems can be used, for example, as a network intrusion detection system and can detect a wide variety of attacks reliably and efficiently when compared to the traditional network intrusion detection systems. In this layered framework, we use a number of separately trained and sequentially arranged sub systems in order to decrease the number of false alarms and increase the attack detection coverage. In particular, our layered framework has the following advantages: The framework is customizable and domain specific knowledge can be easily incorporated to build individual layers which help to improve accuracy, Individual intrusion detection sub systems are light weight and can be trained separately, Different anomaly and hybrid intrusion detectors can be incorporated in our framework, this framework not only helps to detect an attack but it also helps to identify the type of attack. As a result, specific intrusion response mechanisms can be initiated automatically thereby reducing the impact of an attack and this framework is scalable and the number of layers can be increased (or decreased) in the overall framework.

Fig. 2 represents our framework for building Layer based Intrusion Detection Systems (LIDS). The figure represents an '*n*' layer system where every layer in itself is a small intrusion detection system which is specifically trained to detect only a single type of attack, for example the DoS attack. A number of such sub systems are then deployed sequentially, one after the other. This serves dual purpose; first, every layer can be trained with only a small number of features which are significant in detecting a particular class of attack. Second, the size of the sub system remains small and hence, it performs efficiently.

A common disadvantage of using a modular approach, similar to our layered framework, is that it increases the communication overhead among the modules (sub systems). However, this can be easily eliminated in our framework by making every layer completely independent of every other layer. As a result, some features may be present in more than one layer. Depending upon the security policy of the network, every layer can simply block an attack once it is detected without the need of a central decision maker. A number of such layers essentially act as filters, which blocks anomalous connection as soon as they are detected in a particular layer, thereby providing a quick response to intrusion and simultaneously reducing the analysis at subsequent layers. It is important to note that a different response may be initiated at different layers depending upon the class of attack the layer is trained to detect.

The amount of audit data analysed by the system is more at the first layer and decreases at subsequent layers as more and more attacks are detected and blocked. In the worst case, when no attacks are detected until at the last layer, all the layers have the same load. However, the overall load for the average case is expected to be much less since attacks are detected and blocked at every subsequent layer. On the contrary, if the layers are arranged in parallel rather than in a sequence, the load at every sub system is same and is equal to that of the worst case in the sequential configuration. Additionally, the initial layers in the sequential configuration can be replicated to perform load balancing in order to improve performance.

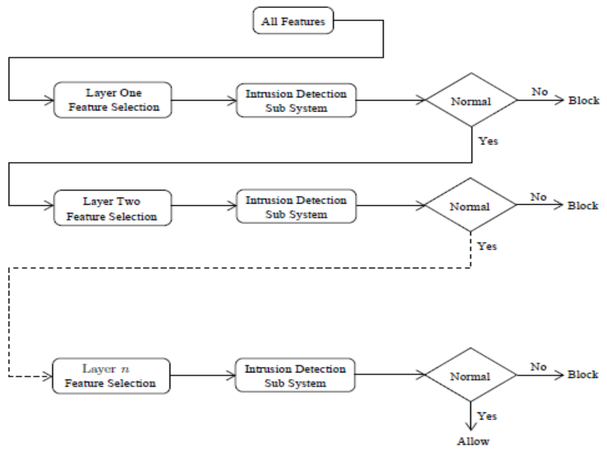
All Features
Layer One Feature Selection
Intrusion Detection Sub System
Normal
No
Block
Yes
Layer Two Feature Selection
Intrusion Detection Sub System
Normal
No
Block
Yes
Layer n Feature Selection
Intrusion Detection Sub System
Normal
No
Block
Yes
Allow

Fig. 2: Layered Structure for building IDS

A. *Components of Individual Layers*

Given that a network is prone to a wide variety of attacks, it is often not feasible to add a separate layer to detect every single attack. However, a number of similar attacks can be grouped together and represented as a single attack class. Every layer in our framework corresponds to a sub system which is trained independently to detect attacks belonging to a single attack class. As a result, the total number of layers in our framework remains small. For example, both, 'Smurf' and 'Neptune' result in *Denial of Service* and, hence, can be detected at a single layer rather than at two different layers.
Additionally, the layered pattern is very general and the number of layers in the overall system can be adjusted depending upon the individual requirements of the network in concern. Consider for example, a data repository which is a replica of a real-time application data and which does not provide any online services. To ensure security of this data, the priority is to simply detect network scans as opposed to detecting malicious data accesses. For such an environment, only a single layer which can reliably detect the *Probe* attacks is sufficient. Hence, the number of layers in this framework can be easily customized depending upon the identified threats and the availability of resources. Even though the number of layers and the significance of every layer in this structure depend upon the target network, every layer has two significant components:

1) Feature Selection Component: In order to detect intrusions, a large number of features can be monitored. These features include 'protocol', 'type of service', 'number of bytes from source to destination', 'number of bytes from destination to source', 'whether or not a user is logged in', 'number of root accesses', 'number of files accessed' and many others. However, to detect a single attack class, only a small set of these features is required at every layer. Using more features than required makes the system inefficient. For example, to detect *Probe* attacks, features such as the 'protocol' and 'type of service' are significant while features such as 'number of root accesses' and 'number of files accessed' are not significant.

2) Intrusion Detection and Response Sub System: The second component in every layer is the intrusion detection and response unit. To detect intrusions, our framework is not restrictive in using a particular anomaly or hybrid detector. A variety of previously well known intrusion detection methods such as the naive Bayes classifier, decision trees, support vector machines and others can be used. Once an attack is detected, the response unit can provide adequate intrusion response depending upon the security policy.

V. Advantages of Layered Structure

- Using our layered framework improves attack detection accuracy and the system can detect a wide variety of attacks by making use of the domain specific knowledge.
- Our framework is easily customizable and the number of layers can be adjusted depending upon the requirements of the target network.
- Our framework is not restrictive in using a single method to detect attacks. Different methods can be seamlessly integrated in our framework to build effective intrusion detectors.
- Our framework has the advantage that the type of attack can be inferred directly from the layer at which it is detected.

VI. Future Work

Every network and application is custom designed and it becomes extremely difficult to develop a single solution which can work for every network and application. In this paper, we described framework and developed methods which perform better than previously known approaches. In order to access application data, a user has no option but to access the application which interacts with the application data. Hence, application access and the corresponding data accesses are highly correlated. In order to detect attacks effectively, we aim to capture this correlation between the application access and the corresponding data accesses. Unified Logging Framework efficiently integrates the application and the data access logs. Unified logging framework can capture user-application and application-data interactions which are significant to detect application level attacks.

If we integrate both models Layered Framework for Intrusion Detection and Unified Logging Framework for Audit Data Collection we can built a attack detection system. By using the unified log, we can capture the user-application and the application data interaction in order to improve attack detection. Further, this interaction is fixed and does not vary overtime as opposed to modelling user profiles which changes frequently. This integrated framework is application independent and can be deployed for a variety of applications.

VII. CONCLUSION

In this paper, we explored the Layered framework which can be used to build efficient intrusion detection systems. In addition, the framework offers ease of scalability for detecting different variety of attacks as well as ease of customization by incorporating domain specific knowledge. The framework also identifies the type of attack and, hence, specific intrusion response mechanism can be initiated which helps to minimize the impact of the attack, Unified logging framework can capture user-application and application-data interactions which are significant to detect application level attacks. The framework is application independent and can be used for a variety of applications. Additionally, the system is robust and can effectively detect disguised attacks.

References

[1] Kapil Kumar Gupta, Baikunth Nath, Kotagiri Ramamohanarao, and Ashraf Kazi. Attacking Confidentiality: An Agent Based Approach. In Proceedings of IEEE International Conference on Intelligence and Security Informatics, pages 285–296. Lecture Notes in Computer Science, Springer Verlag, Vol (3975), 2006.

[2] Jonnalagadda Sravan Kumar and Subha Sree Mallela, Reducing false alerts using intelligent hybrid systems. (IJCSIS) International Journal of Computer Science and Information Security,pages 291-297, Vol. 9, No. 5, May 2011.

[3] Peyman Kabiri and Ali A. Ghorbani. Research on Intrusion Detection and Response: A Survey. International Journal of Network Security, 1(2):84–102, 2005.

[4] B Mukherjee, L Todd Heberlein, K N Levitt, 1994. "Network intrusion detection. IEEE Network, Vol. 8, No. 3, pp.26–41, 1994.

[5] H. S. Teng, K. Chen and S. C. Lu. "Security Audit Trail Analysis Using Inductively Generated Predictive Rules". In Proceedings of the 11th National Conference on Artificial Intelligence Applications, pages 24-29, IEEE, IEEE Service Center, Piscataway, NJ, March 1990.

[6] P A Porras. "STAT: A State Transition Analysis Tool for Intrusion Detection". Master's Thesis, Computer Science Dept., University of California, Santa Barbara, July 1992.

[7] T Dunigan and G Hinkel, "Intrusion detection and intrusion prevention on a large network: A case study", Proc. of workshop on intrusion detection and network monitoring, 1999.

[8] K Ilgun, "USTAT: A Real-time Intrusion Detection System for UNIX," 16-28. Proceedings of the 1993 Computer Society Symposium on Research in Security and Privacy. Oakland, California, May 24-26, 1993. Los Alamitos, CA: IEEE Computer Society Press, 1993.

[9] T. D. Garvey and T. F. Lunt, "Model based intrusion detection". In Proceedings of the 14th National Computer Security Conference, pages 372-385, October 1991.

[10] S. Kumar. "Classification and Detection of Computer Intrusions". PhD Thesis, Department of Computer Science, Purdue University, August 1995.

[11] W. Lee and S. Stolfo and K. Mok. "A Data Mining Framework for Building Intrusion Detection Models". In Proceedings of the IEEE Symposium on Security and Privacy, 1999.

[12] M Debar, DBecke, and A Siboni. "A Neural Network Component for an Intrusion Detection System". Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy, 1992.

[13] K. Shah, N. Dave, S. Chavan, S. Mukherjee, A. Abraham and S. Sanyal, "Adaptive Neuro- Fuzzy Intrusion Detection System", IEEE International Conference on Information Technology: Coding and Computing (ITCC'04), USA, IEEE Computer Society, Volume 1, pp. 70-74, 2004.

[14] S. Mukkamala, A.H. Sung, A. Abraham and V. Ramos, "Intrusion Detection Systems Using Adaptive Regression Splines", 6th International Conference on Enterprise Information Systems, ICEIS'04, Portugal, I. Seruca, J. Filipe, S. Hammoudi and J. Cordeiro (Eds.), ISBN 972-8865-00-7, Vol. 3, pp. 26-33, 2004.

[15] N.B. Amor, S. Benferhat, and Z. Elouedi, "Naive Bayes vs. Decision Trees in Intrusion Detection Systems," Proc. ACM Symp. Applied Computing (SAC '04), pp. 420-424, 2004.

[16] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur, "Bayesian Event Classification for Intrusion Detection," Proc. 19th Ann. Computer Security Applications Conf. (ACSAC '03), pp. 14-23, 2003.

[17] Y.-S. Wu, B. Foo, Y. Mei, and S. Bagchi, "Collaborative Intrusion Detection System (CIDS): A Framework for Accurate and Efficient IDS," Proc. 19th Ann. Computer Security Applications Conf. (ACSAC '03), pp. 234-244, 2003.

Subha Sree Mallela pursued her M. Tech in Computer Science and Engineering at Acharya Nagarjuna University. Now, she is an Assistant Professor in department of Computer Science in DMSSVH College of engineering, Machilipatnam (India). Her current research interest includes Network Security, Software Engineering and Aspect Oriented Programming.

Sravan Kumar Jonnalagadda pursued his M. Tech in Computer Science at Acharya Nagarjuna University. Now, he is an Assistant Professor in department of Information Technology in DMSSVH College of engineering, Machilipatnam (India). His current research interest includes Network Security and Software Engineering.