Agent-Based Distributed Intrusion Alert System

Arjita Ghosh and Sandip Sen
{arjita-ghosh,sandip}@utulsa.edu

University of Tulsa, Tulsa OK 74104, USA

Abstract. Intrusion detection for computer systems is a key problem in today's networked society. Current distributed intrusion detection systems (IDSs) are not fully distributed as most of them centrally analyze data collected from distributed nodes resulting in a single point of failure. Increasingly, researchers are focusing on distributed IDSs to circumvent the problems of centralized approaches. A major concern of fully distributed IDSs is the high false positive rates of intrusion alarms which undermine the usability of such systems. We believe that effective distributed IDSs can be designed based on principles of coordinated multi-agent systems. We propose an Agent-Based Distributed Intrusion Alert System (ABDIAS) which is fully distributed and provides two capabilities in addition to other functionalities of an IDS: (a) early warning when pre-attack activities are detected, (b) detecting and isolating compromised nodes by trust mechanisms and voting-based peer-level protocols.

1 Introduction

Work on securing networks and hosts from malicious attackers have concentrated on two areas: i) Intrusion prevention mechanisms that include cryptographic techniques to safeguard sensitive information from unauthorized access and manipulation, ii) Intrusion detection mechanisms that recognize an ongoing attack and respond appropriately to thwart such intrusive, disruptive behaviors. Over the last decade, research in the latter of this two approaches has been leaning towards a distributed framework to circumvent the demerits of centralized intrusion detection systems(IDS), e.g. [9]. But currently available distributed intrusion detection system (DIDS) are not fully distributed in design: they collect data from distributed nodes but analyze them centrally. In this framework the first problem is the transfer of raw data which can lead to security breaches. The second drawback is that an intruder can take control over the whole network in a if it can compromise the central server. Hence, it is preferable to have distributed IDS. However, several problems with current distributed IDSs, e.g., the high rate of false positive rates, insufficient protection against compromised nodes, etc. prevent them from being deployed.

We are interested in enhancing the mechanism of distributed intrusion detection with a layer of active, vigilant, monitoring defense mechanism. This layer will detect precursive activities to actual attacks with the help of a large number of distributed, loosely-coupled, computational units. We believe the requirements

of this layer can be effectively met by a distributed computational framework. In this paper, we discuss our proactive agent-based early warning system that uses coordinated surveillance by incorporating inter-agent communication and distributed computing in decision making to identify early signs of attack and recognize situations that are likely to predate an actual attack (e.g., systematic scanning activity) and alert the system administrator. Our ABDIAS system has two primary goals:

Early, distributed threat detection: We want our system to be an early warning system that, in addition to its ability to respond to ongoing attacks, should be able to alert the system administrator to signs of pre-attack activities. By local monitoring and sharing individual belief-estimates, agents can recognize and preempt security threats and pre-attack activities, thus responding to attacks before the system is endangered.

Effectively handle compromised nodes: To address a key problem in current distributed IDSs, we want our system to be capable of detecting and isolating compromised nodes. We tackle this problem by incorporating trust mechanism including the application of majority voting among agents in neighborhood.

To effectively model the inherent domain and environmental uncertainty that must be handled by IDSs, our agents represent their knowledge about the possible attack scenarios in the form of Bayesian networks[5]. This knowledge is derived offline from analysis of repositories of network attack related data. Our research emphasis is to distribute this knowledge such that each agent is required to monitor relatively few aspects of the local network neighborhood but, together, the system is still able to reliably detect security threats through timely coordination with agents. To enable such distributed inference, we use multiply-sectioned Bayesian Networks [19] for representing domain knowledge and clique-tree propagation algorithm [7] for reasoning. To reduce network congestion, we group agents into localities and then design the intrusion detection protocol to limit the significant majority of communications within localities.

The organization of this paper is as follows: Section 2 describes the background of this work, Section 3 outlines the distributed agent-based IDS, Section 4 depicts the distributed intrusion detection mechanism, Section 5 presents the trust model among peer nodes, Section 6 shows the experimental framework and result, and section 7 summarizes our work and discusses future scope.

2 Background

2.1 Intrusion Detection Systems

The process of monitoring events occurring in a computer system or network and analyzing them for sign of intrusions is known as *Intrusion Detection*. Based on data source Intrusion Detection Systems (IDS) can be characterized as follows:

Host based: uses system call data from an audit process that tracks all system calls made on behalf of each user on a particular machine.

Multihost based: collects audit data from multiple hosts and analyzes them centrally to detect intrusions.

Network based: typically uses network traffic data from a network packet sniffer,e.g., TCP-dump, along with audit data from one or more hosts.

Host-based IDSs examine the activities on a specific host. This allows them the advantage of having greater access to the logs and files of a particular computer, while being limited in what external activities they can see. Network-based IDSs placed into a sensor on a segment see only the traffic of that segment. To circumvent these problems several researchers focus on the field of multihost based IDSs or distributed IDSs. Besides the above categorization, IDSs can be classified into two types based on the model of intrusions, namely *misuse intrusion detection* and *anomaly intrusion detection*.

Misuse detection model: detects intrusions by identifying activities that correspond to known intrusion techniques or system vulnerabilities. It uses well-defined attack patterns to identify intrusions. Its merit is relatively low false positive rate. But it can detect only well-known attacks, leaving vulnerable to new attacks.

Anomaly detection model: detects intrusions by identifying activities distinct from a user's or system's normal behavior. Its demerits are i) high false positive rates which makes the system unreliable, and ii) a susceptibility to being compromised when malicious activities disguise as acceptable behaviors.

2.2 Current Distributed IDSs

AID (Adaptive Intrusion Detection system) is a multihost based misuse detection system that proposes a client-server architecture consisting of a central monitoring station and several agents on the monitored hosts. The audit data collected by agents are transferred to the central monitoring station, buffered in a cache and analyzed by a real-time expert system. AID is being developed at Brandenburg University of Technology at Cottbus, Germany.

AAFID (Autonomous Agents For Intrusion detection)[16] is a distributed anomaly detection system that employs autonomous agents at the lowest level for data collection and analysis. At the higher levels of the hierarchy transceivers and monitors are used to obtain a global view of activities.

DIDS (Distributed Intrusion Detection system)[11] is the first intrusion detection system that aggregates audit reports from a collection of hosts on a single network. The architecture consists of a host manager, a monitoring process or collection of processes running in background.

EMERALD (Event Monitoring Enabling Responses to Anomalous Live Disturbances)[10] is an analysis and response system that is able to address unanticipated misuse in large network-based enterprises, within an interoperable and scalable modular system framework.

CSM (Cooperating Security Managers)[17] is designed for a distributed network environment. The goal of CSM is to detect intrusive activities in a distributed environment without the use of a centralized director.

AID, *AAFID*, *DIDS* and *EMERALD* have tried to implement the concept of distributed intrusion detection either by capturing misuse or anomaly or both.

But their common drawback is that each of them is, to a varying degree, hierarchical in nature. Hence, raw data must be transferred up to the highest level to be analyzed, which gives opportunity to the intruder to intercept the data. Moreover, the analysis of this huge amount of data consumes a significant amount of time and resources, creating system bottlenecks. CSM presents a different approach that makes the system fully distributed in nature. We adopt the same approach and enhance the functionality by differing from CSM in the following aspects: allowing the agent to pass their decision (not the raw data) to neighboring agents (not to everyone). It is safer since a malicious entity cannot get raw data and is quicker because decision is made within a local neighborhood of the hosts being monitored. Another novel characteristic of our system is to identify a compromised agent by incorporating trust mechanism and majority voting.

3 ABDIAS Architecture

We envisage a distributed, lightweight agent-based intrusion alert mechanism that leverage of our previous experience with developing coordinated multiagent systems[13, 14]. In this approach we view our agents as autonomous, reflexive, proactive and cooperative entities. They are responsible for collecting data, analyzing them and making appropriate inference from the analysis. Their inference process uses the collected data as evidence in the Bayesian network. Monitoring and analysis work is duplicated for accuracy and fault tolerance, e.g., handle the problem of some agents getting compromised.

In this paper we present an architecture (see Figure 1) where several autonomous agents form a layer of defense that surrounds the internal system network. Agents are grouped into several islands/neighborhoods, N_i, inside this layer to focus communication within the neighborhood[1]. Inter-neighborhood communication is used only when consensus cannot be arrived within a neighborhood. Agents are responsible for alerting system/network administrator with any possible attack-related activities. Agents are cognizant of a Bayesian network model of the structure of well-known attack types and as well as normal usage pattern which is constructed offline from data repositories containing system logs from ongoing attacks[2]. This network has been partitioned into multiple sub-nets based on the spatial location of the agents. And each of these subnets is common knowledge to agents in same neighborhood. Each agent performs a certain predefined security monitoring function at a peer host. In addition, some agents are responsible for monitoring network traffic data for possible network attack signatures. Agents may collect data locally, i.e., from audit trail, or over the entire network, i.e. from TCPdump data.

[1] Our restriction of communication refers only to the communication for the purpose of intrusion detection, and does not restrict normal communication between nodes.

[2] It is easy to confuse the term "network" between the physical network hosting the system and the Bayesian network model of intrusion detection. Unless otherwise specified, we will use *network* and related terms to refer to the Bayes net model.

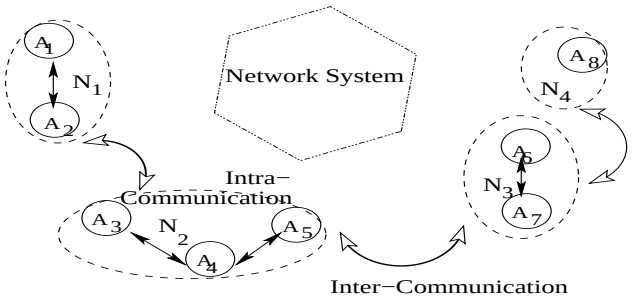

Fig. 1. ABDIAS: Architecture

We use Bayesian networks to represent existing knowledge of different security threats [4]. A Bayes net captures the mutual influence of different domain variables on target attributes. Using a Bayes net model we can infer the probabilities of the occurrence of different intrusion types, which are easy for human security investigators to interpret. Moreover, this representation can easily accommodate prior domain knowledge. We believe that the Bayesian net representation is the best currently available approach for handling intrinsic uncertainties in intrusion detection. A Bayes net based approach also allows for combining competing intrusion detection methods such as anomaly detection and signature recognition [12]. To facilitate this, we generate one Bayes net whose target node classifies several known attack types and normal system behavior. Using this agents can detect either normal behavior or a known attack type. If none of the probability of target nodes, given input feature values, cross the threshold, anomalous behavior is suspected.

The Bayes net attack model is updated by distributed incremental inference process based on dynamically arriving incomplete information. Such local model updates may necessitate coordination with peers in the neighborhood and, more infrequently, inferences will be communicated to peers in different neighborhoods. An agent can request others to re-confirm their inference and also request updates about network properties that are not monitored locally. If the requesting agent, R, suspects a problem with the information received from another peer, because of inconsistencies with local inference, the agent may check the status of the sender agent, S. If such a suspicion is raised, R can poll its neighborhood agents, using a majority voting mechanism, to confirm the suspicion. On confirmation, steps are taken to tag the sending agent as a possibly compromised node, and then isolate it from the network. If the neighborhood agents fail to confirm the suspicion and does not provide a convincing argument, R, can ask for confirmation from agents beyond its immediate neighborhood. Thus by introducing distributed trust mechanisms and majority voting, we make our system robust against situations where a security breach has comprised some nodes.

Figure 2 depicts the ABDIAS agent architecture. The agent tasks are:
Data Collection and analysis: Each agent collects audit or network data from

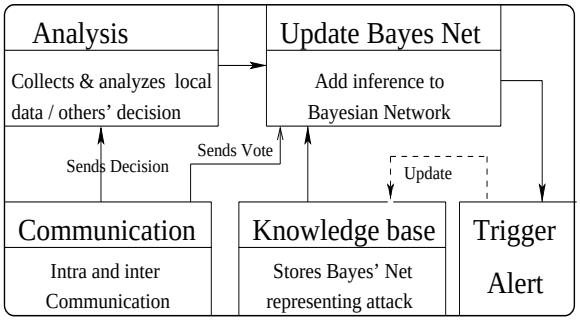

Fig. 2. Architecture of an agent embedded in a node

its own domain, analyzes them and draws inference from the analysis.
Network update: Agent updates the belief associating the node and distributes its belief to all neighboring nodes.
Communication: The inference is passed to peers either in same locality or greater neighborhood. If any agent is not satisfied with the received inference, it may ask the sender to verify its inference. An agent may also ask other agents, including those from a different neighborhood, to vote on the inference submitted by another agent to evaluate whether this latter agent has been compromised.
Trigger Alert: When an agent recognizes that target node exceeds threshold of one known attack, it triggers an alert for that particular attack, and communicates it to the system administrator. Besides this misuse detection, we allow the agents to trigger an alert representing "something is wrong" when the activated target node does not belong to those representing normal behavior or any of the known attack types considered while creating Bayes net. The administrator can confirm the attack and take necessary responses, or otherwise reject the alert. Thus our agents can detect both anomaly and misuse intrusions.
Update Knowledge base: If the system administrator confirms an anomaly alert, a Bayes net is modified to accommodate this new attack in the knowledge base which will allow this attack to be recognized as a known attack in future. Thus our system is adaptive to the discovery of new types of network intrusions.

4 Forecasting attack using Bayesian Hypothesis

4.1 Bayesian Network

A Bayesian network (BN) is a graph-based modeling approach to represent dependencies among domain variables [5]. A BN is a directed acyclic graph with nodes representing the variables and each directed edge representing a dependency between the corresponding variables. The effect of the parents of a node on a node is represented by conditional probabilities of that variable given values of its parent nodes in the form of a conditional probability table (CPT). We represent attack structures as Bayesian network for the following reasons:

• BNs can readily handle incomplete data sets. Agents in ABDIAS have limited view of the network and may receive only partial information about an attack.
• BNs can represent causal relationships which can help IDSs predict the consequences of intervention by combining a priori knowledge and observed data.
• BNs allow updating of the belief or the probability of occurrence of the particular event for the given causes and several networks can be used by IDSs to recognize the possibility of new attacks.

In ABDIAS, a Bayes tree is first learned from a database of known attacks. This tree is then partitioned into several subtrees following the principle of Multiply Sectioned Bayesian Networks (MSBNs) [19], and distributed among agents. Though Bayes trees have been used for centralized IDSs, we are not cognizant of any work in the field of computer security where agents combine distributed local inference from partitioned trees to form a proactive distributed IDS.

4.2 Inference with Multiply Sectioned Bayesian Networks

A Multiply Sectioned Bayesian Network (MSBN) consists of interrelated subnets each of which encode an agent's knowledge of a sub-domain. Exact, distributed probabilistic inference can be performed using MSBNs: a perfect match of for our ABDIAS requirements where each agent have knowledge of only a sub-domain.

Existing methods for inference in MSBNs are extensions of a class of methods for inference in single-agent Bayesian networks: message passing in junction trees [5]. We have used the *exact* Bayesian network inference algorithm [3], namely, *linked junction forest* (LJF) method [19] on MSBN. It first transforms each subnet of a multiply connected network (here, a Bayes tree) into a clique tree or junction tree by clustering the triangulated moral graph of the underlying undirected graph, and then performs message propagation over the clique tree. Inter-neighborhood (i.e., between two junction trees in Linked Junction Forest (MSBN)) message passing is performed through a *linkage tree* between a pair of peer nodes in adjacent neighborhood. Though exact belief update is NP-hard [2, 15] we can still use it for IDS domains as the subnet sizes are manageable.

In our architecture, agents in the same neighborhood first find the junction tree of associated Bayes sub-tree (discussed in Section 6). Each agent monitors one/more cluster(s) and when they find any evidence, they update the belief tables at that node, and pass the message to neighboring agents for updating their tables. The clique propagation algorithm (part of *LJF* algorithm) works efficiently for sparse networks. Our approach exploits the structure of the problem to gain efficiency in computing *exact* probabilities. As we divide our entire Bayes network into several subparts, the clique-size of subnets is not large, and hence inferences can be made in real-time.

Figure 3 shows a MSBN. We denote by N_i the group of agents $A_1, \ldots, A_k$ whose knowledge is encoded in subnet D_i. The peers of each N_i can only observe locally. Once a multiagent MSBN is constructed, agents may perform probabilistic inference by computing the query $P(x|e)$, where x is any variable within the sub-domain of a group of agents and e denotes the observations made by all agents. The key computation is to communicate the impact of observations to

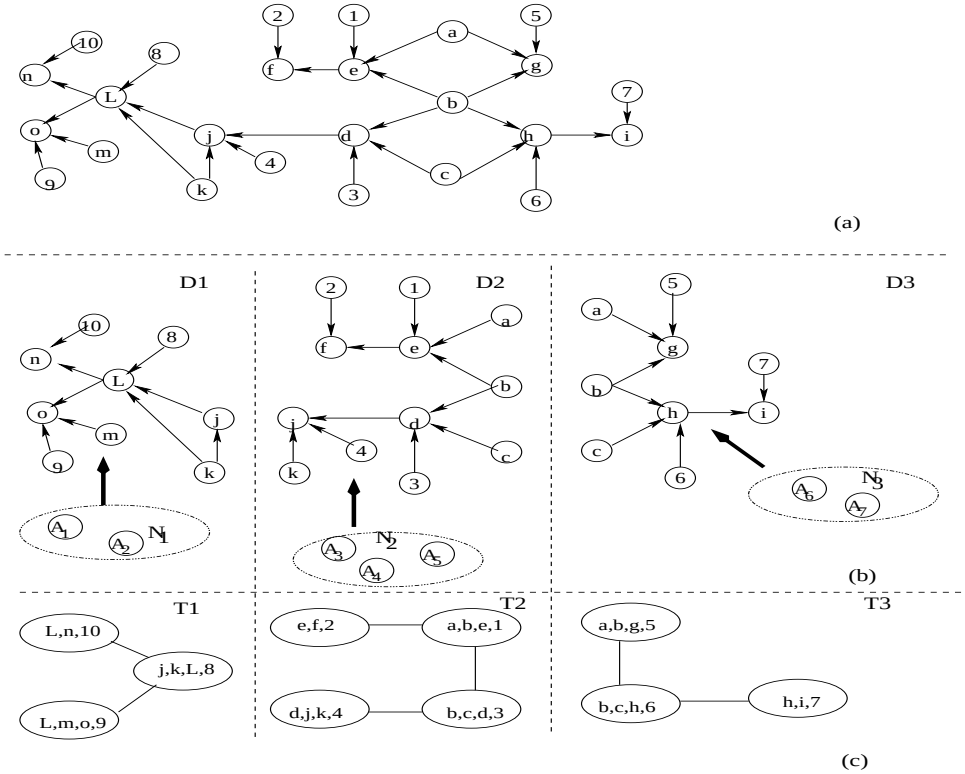

Fig. 3. (a) A Bayes network (b)The DAGs of three subnets of a MSBN; each DAG is under supervision of each group of agents (c) JTs converted from the subnets

all agents. Often agents in N_i computes the query $P(x|e_i, e_i')$, where e_i is the local observations made by N_i and e_i' is the observations made by agents of other groups up to the latest communication. Each subnet, D_i, formed by a group of peer nodes may be multiply connected. To facilitate exact inference with message passing, the LJF method compiles each subnet into a JT, called a local JT, and converts each d-sepset (analogous to separator in JT) into a JT, called a linkage tree. Figure 4 illustrates the three local JTs and two linkage trees of the original Bayes network. Since reduction of network congestion is a desirable goal, we chose the LJF method as it has been shown that among different distributed multiagent inferences in MSBNs, LJF requires least amount of system-wide messages during communication [18].

5 Trust among peers

We now present the use of trust mechanism among peer agents, a relatively novel concept in computer security, but in our estimation a key, integral feature

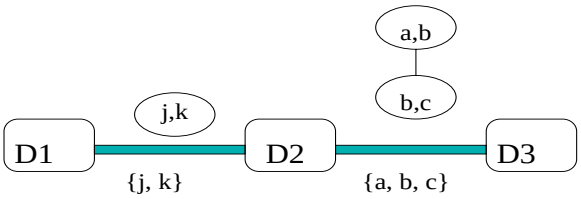

Fig. 4. Linkage trees for JTs of Figure 3

of any distributed IDS. Our prior research in multiagent systems have shown the pros and cons of believing others [14]. In ABDIAS, each agent is limited in the things it can monitor. Hence each agent needs to rely on other agents for non-local data. As intruders will try to breach any network node, agents must continually monitor their trust on other agents to quickly identify compromised nodes in the system. While compromised nodes inside a locality may attempt to influence the JT, such activities also provide detection clues. If any agent finds that received messages from an agent does not tally with the anticipated trends, it may take any or all of the actions: i) ask sender to resend the message, ii) check its own belief matrix with that of the sender, iii) ask for vote to identify and subsequently eliminate the suspicious agent from the system.

Each agent, i, in ABDIAS maintains a belief table, B_i about other agents, which it updates whenever it receives any message. The entry for the jth agent, is dependent on the nature of the last m messages received from that agent, and is defined as $B_i[j] = \frac{S_j - R_j}{m}$, where S_j and R_j is respectively the number of messages coming from agent j that does or does not match with agent i's anticipation. When agent i is not satisfied with agent j's response for corroborating an inference, it checks its belief matrix entry for the sender agent. If the corresponding entry is less than a threshold, i asks other neighborhood agents to vote on j's trustworthiness. When an agent is asked for vote on another peer, it checks its belief table for its trust in that peer and votes for or against the agent if it finds the value above or below, respectively, of the trust threshold. If the vote difference between total for and against votes is small, i asks vote from agents in a larger neighborhood. If the candidate agent supervises a node in the linkage tree, it has connections with remote agents whose vote can be used to break the tie. If a tie still exists, the local group takes the decision randomly. Such a procedure is robust in removing not only single, but multiple, compromised nodes from the system.

Each agent in one neighborhood knows the structure of subnets and CPTs associated with each node. If the probability of a particular node given specific states of its parents, was gradually increasing but this observation is countered by another agent, the latter's response is marked as inconsistent.

Thus ABDIAS can detect compromised nodes and maintain its performance, reliability and availability to users which makes ABDIAS resilient in the situation where the intruder has already captured some nodes. We believe, this robustness is unique to ABDIAS among Distributed IDSs.

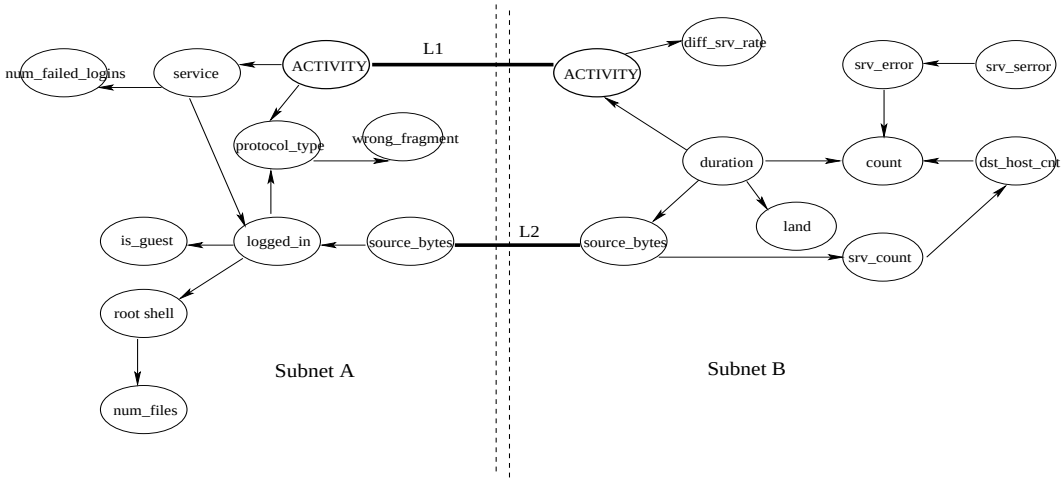

Fig. 5. Subnets A and B are locally computed in two neighborhoods. L1 and L2 depict the connection between them. ACTIVITY is the target node; it has 5 classifications: 4 attacks and 1 normal situation

6 Experimental setup and Results

We used the KDD Cup 1999 Intrusion detection contest data [6] in our experiments. This data was prepared by the 1998 DARPA Intrusion Detection Evaluation program by MIT Lincoln Labs [8]. Lincoln labs set up an environment consisting of a local-area network simulating a typical U.S. AirForce LAN. They acquired nine weeks of raw TCPdump data that was processed into connection records. The original data contains 744MB of data with 4.94 million records. The dataset has 41 attributes for each connection record plus one class label specifying one of 24 attacks or normal condition. All these attacks fall into four major categories: i) Denial of Service (DoS), ii) Remote to User (R2U), iii) User to Root (U2R) and iv) Probing (Probe).

We have also processed the original data from 24 attacks to the above-mentioned four major attack classes. The dataset for our experiments contain 11800 records, randomly selected from the original dataset. This dataset has five different classes: 4 attacks and 1 normal. We ensured that the number of data instances selected from from each class was proportional to their frequency in the original data set. The only exception is that all instances of classes with significantly small frequency are selected. This dataset is used to generate the diagnostic Bayesian network. We have used 17 of the 41 attributes as these are considered the most important variables for intrusion detection by researchers in this field. Bayesian Network Power Constructor (BNPC) [1] has been used to generate Bayes network from this dataset of 11800 records and 17 attributes. Then we section into two subnets as described in figure. 5 maintaining the rules for sound sectioning in MSBN literature [19]. Thereafter we used LJF method [19]

	Training data: 11800 records		Test data: 15000 records	
Activity Type	USBN	MSBN	USBN	MSBN
Normal	96.67%	94.37%	97.11%	95.71%
Probe	92.37%	89.50%	92.37%	88.12%
DoS	91.85%	90.78%	91.20%	89.30%
U2R	85.00%	84.70%	81.00%	79.08%
R2U	89.16%	86.07%	85.43%	82.73%

Table 1. Performance of USBN and MSBN on DARPA Intrusion detection data set.

to detect intrusions. To test our network, a new dataset consisting of 37 attacks has been considered. Some of these attacks do not come under the four major attack classifications mentioned above, in which case ABDIAS detects it as a new attack or anomalous behavior of the network. Table 1 compares the detection rate for each activity classification (4 attacks and 1 normal) between unsectioned BN (USBN) and MSBN on training and test data. Though a centralized IDS, using USBN, can be seen to have performed better, performance of ABDIAS, using MSBN, is encouraging given its fully distributed nature.

7 Conclusion

This paper presented ABDIAS, a peer level distributed intrusion alert system. Using distributed computation and message passing between distributed agents, ABDIAS recognizes pre-attack activities in the system and can alert the system administrator for taking appropriate actions. The two novel feature of ABDIAS is (a) the ability to form completely distributed inference based on partitioned Bayesian networks to provide early warning of possible future attack, and (b) recognition of compromised nodes by peer-level collaboration.

ABDIAS improves on existing distributed IDSs by using distributed monitoring and diagnosis techniques for early detection of imminent attacks and compromised nodes in the network. The use of Bayesian networks enable our system to track misuse as well as anomalous behavior in the system. Estimation of trust by applying majority voting makes the system intrusion-tolerant.

Currently, we ran experiments considering one USBN having n-ary classification (here, 4 attacks and 1 normal situation). In future we want to compare the performance of ABDIAS considering one BN for each attack type and performing binary classification. The false positive rate of ABDIAS is low, but needs to be compared with other DIDSs. We also want to enhance the system so that it can update its knowledge base with newly discovered attack which has been recognized as an anomalous situation.

Acknowledgments: This work has been supported in part by NSF award IIS-0209208.

References

1. J. Cheng, D. Bell, and W. Liu. Learning bayesian networks from data: An efficient approach based on information theory. Technical report, University of Alberta, Canada, 1998.
2. G. F. Cooper. The computational complexity of probabilistic inference using bayesian belief networks. *Artificial Intelligence*, 42:393–405, 1990.
3. H. Guo and W. H. Hsu. A survey of algorithms for real-time bayesian network inference. In *AAAI/KDD/UAI-2002 Joint Workshop on Real-Time Decision Support and Diagnosis Systems*, Edmonton, July 2002.
4. M. Y. Huang and T. M. Wicks. A large-scale distributed intrusion detection framework based on attack strategy analysis. In *Web proceedings of the First International Workshop on Recent Advances in Intrusion Detection*, Louvain-la-Neuve, Belgium, September 1998.
5. F. V. Jensen. *An Introduction to Bayesian Networks*. Springer Verlag, New York, NY, 1996.
6. Kddcup 99 intrusion detection data set. URL: http://kdd.ics.uci.edu/databases/kddcup99/kddcup.data_10_percent.gz. DARPA Intrusion data repository.
7. S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their applications to expert systems. In *Proceedings of the Royal Statistical Society, Series B., 50*, pages 154–227, 1988.
8. Mit lincoln laboratory. URL: http://www.ll.mit.edu/IST/ideval/. DARPA Intrusion data repository.
9. B. Mukherjee, T. L. Heberlein, and K. N. Levitt. Network intrusion detection. *IEEE Network*, 8(3):26–41, May/June 1994.
10. P. A. Porras and P. G. Neumann. Emerald: event monitoring enabling responses to anomalous live disturbances. In *20th National Information Systems Security Conference*, October 1997.
11. J. B. S. Snapp and G. D. et al. Dids (distributed intrusion detection system) motivation, archi-tecture, and an early prototype. In *Fourteenth National Computer Security Conference*, Washington, DC, October 1991.
12. S. L. Scott. A bayesian paradigm for designing intrusion detection system. *Computational Statistics and Data Analysis*, 45(1):69–83, February 2004.
13. S. Sen. Developing an automated distributed meeting scheduler. *IEEE Expert*, 12(4):41–45, July/August 1997.
14. S. Sen. Believing others: Pros and cons. *Artificial Intelligence*, 142(2):179–203, 2002.
15. S. Shimony. Finding maps for belief networks is np-hard. *Artificial Intelligence*, 68:399–410, 1994.
16. E. H. Spafford and D. Zamboni. Intrusion detection using autonomous agents. *Computer Networks*, 34(4):547–570, October 2000.
17. G. White, E. Fisch, and U. Pooch. Cooperating security managers: A peer-based intrusion detection system. *IEEE Network*, 10(1):20–23, 1996.
18. Y. Xiang. Comparison of multiagent inference methods in multiply sectioned bayesian networks. *International Journal of Approximate Reasoning.*, 33(3):235–254, August 2003.
19. Y. Xiang, D. Poole, and M. Beddoes. Multiply sectioned bayesian networks and junction forests for large knowledge-based systems. *Computational Intelligence*, 9(2):171–220, 1993.