CHAPTER 6

CLUSTER BASED HYBRID INTRUSION DETECTION SYSTEM WITH FAULT TOLERANCE

6.1 INTRODUCTION

In the distributed intrusion detection method, the agent can travel from one node to a different node within a network and perform the task concurrently and independently in different processes over the distributed system (Snapp et al 1991). Hence, the agent has equal amount of information, and bear equal responsibility to take the final decision. However, a single server needs to have enough memory space and processing power in the centralized procedure. Hence, the centralized procedure is resource intensive. Moreover, the distributed procedure is more reliable than the centralized procedure due to the absence of single point of failure. Also, they are easily scalable. The data is received from the static agent and the mobile agent to reduce the network bandwidth usage by moving the data analysis computation to the location of the intrusion data.

Nevertheless, despite all its advantages, the distributed generalized intrusion detection procedure has the following limitations. The distributed procedure has considerably reduced the data forensic capabilities and over the years, they exchange more messages than the centralized procedure. Hence, the communication cost of distributed procedure is very high. For instance, the most efficient distributed procedure proposed by (Kannadiga & Zulkernine
2005) has added the concept of mobile agent with distributed intrusion detection environment.

Most of the earlier distributed procedures have produced gaps in data analysis when the system is offline. However, they need more time or messages to detect intrusion.

It became apparent that many distributed procedures have determined only real time alerts. Most of the existing distributed procedures rely on producing a real time response for intrusion detection. In contrast, the procedure proposed by (Kannadiga & Zulkernine 2005) has built the mobile agent between the host and network to resolve the intrusion detection as in the centralized procedure. Because of all the above reasons, there is a need for a new distributed procedure that can reduce the network bandwidth and performance degradation on the target.

An agent in a node can either be an active or a reactive component during network implementation. Links and nodes are reactive components that react to incoming packets and apply their behavior to the packet. Compared to them, agents are active components. For instance, when transport agents, Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) react to external comments for sending data, they may generate the necessary packets for data transmission or generate connection control packets (Roshan & Leary 2004).

This chapter presents a new distributed procedure, Hybrid intrusion detection of cluster based network with fault tolerance to detect anomaly detection and policy detection. It uses mobile agent with fault tolerance techniques to eliminate the gaps in data analysis when the system is offline and it has the ability to operate in a heterogeneous environment. This chapter is organized as follows. Section 6.2 describes the underlying computational
model. Section 6.3 discusses the principle behind the proposed procedure, and Section 6.4 explains the intrusion detection mechanism in the proposed procedure. In Section 6.5, the performance of the proposed procedure is analyzed and compared with the existing distributed procedures. Finally, Section 6.6 contains the concluding remarks.

### 6.2 SYSTEM ARCHITECTURE

The system consists of 3 layers, where each layer has a unique identity. The processes communicate through a logical communication channel by message passing. There is no shared memory in the system. The messages are delivered at the destination in the same order as sent by the sender, with arbitrary but finite delay. The messages are neither lost nor duplicated and the entire system is fault-free. The layers in the system are classified into network and mobile agent layers, and they send messages to the system administrator using a decision making agent (Siqueira & Abdelouahab 2006). They are further classified into replication agent and profile database. The decision making agent triggers the 6 messages such as REQUEST, REPLY, CANCEL and ACK due to the execution of applications. The replication agent generates the control messages including CALL and REPORT as a result of the execution of the intrusion detection procedure.

When a user first enters into a network, they are asked to REQUEST...
sends alert message to all the hosts and the network. So the agent sends a CANCEL message to all the other agents and sends a message to the administrator to block the authorized user’s activity. These two assumptions are essential to record consistent information of distributed intrusion detection.

Based on the merits of agent technology, this chapter puts forward a distributed intrusion detection system based on agents proposed by (Siqueira & Abdelouahab 2006) and (Kannadiga & Zulkernine 2005) and is applied in the proposed architecture. A distributed system requires a robust communication infrastructure and is based on the client/server model. Individual agents can be relatively simple and homogeneous, their purpose lies in making use of many host computers rather than performing complex sub-tasks themselves. To develop such a massively distributed system, the agent designer should be supplied with a simple method of connecting agents over a network. In addition, failure recovery is of great concern. When an agent fails, the integrity of the data set has to be preserved. The server must have a way of reallocating a subtask to another agent.

The proposed architecture is composed of three layers: Layer 1 consists of host agent and net agent. Layer 2 has the mobile agent and Layer 3 comprises of the decision making and replication agents. The system architecture is given in Figure 6.1. This chapter will describe every layer in this system in detail.

6.2.1 Layer 1 Architecture

Layer 1 encloses the host agent and net agent. Every host agent and net agent in the network has the general intrusion detection and response functions, processes the information and based on this processing makes a decision. A host agent or net agent generates an ID (Intrusion Detection)
event when it detects suspicious activities may or may not be an intrusion. Examples of such activities are suspicious connections, failed login attempts, modification of system sensitive file from suspicious users. The ID events related to such suspicious behavior are sent by the host agent or net agent to the mobile agent in layer 2.

![System Architecture of Layer Based Intrusion Detection System](image)
Host Agent

The vital goal of a host agent is to identify unauthorized users before they get access or during the process of getting access. The task of the host agent is to protect the systems available in its environment. Host agent intrusion detection system to evaluate data that originate from application logs and operating system event logs. These host event logs encompass evidence about file accesses and program performances associated with privileged users. If protected correctly, the event logs may be generated and entered into the database. When suspicious events are not declared as intrusions, the host agent creates an ID event and transmits it to layer2. Every type of ID event denotes a kind of possible attack. For instance, a possible attack could be an authorized user misusing their rights inside the network or a failed login attempt or port scanning or modification to the system files. Host-based intrusion detection systems also analyze user statistics to determine misuse.

Net Agent

The role of the net agent is to detect network intrusions and prevent network intruders from outside sources. Mostly net agents are installed at the entrance of the network or on the server. The net agent controls the network traffic, records all suspected events in a database and responds to intrusions. The way the net agent deals with the intrusions is the same as host agent. In layer 1, the net agent identifies whether the user is authorized or unauthorized. If the right of entry is found to be unauthorized, then the net agent is the one that performs these identifications and concludes that the user is unauthorized. Hence the net agent blocks the user. The unauthorized users are thus blocked using the basic concept of password authentication. The network monitor i.e., the net agent in layer 1 identifies the password pattern from network traffic.
6.2.2 Layer 2 Architecture

Layer 2 consists of mobile agents dispatched by layer 3 for the continuous observation of the host and net agent in layer1. The mobile agent visits all the hosts in the network and collects the related attack data information. Then it correlates the collected information with the data it has received from the other host agent or net agent that has generated the same type of ID event. The MA sends out alerts when it detects intrusions. On one hand it informs every host it has visited to take the corresponding response measures and indicate the alerts; on the other hand, it transmits these related information to layer3 and shows them on the user interface.

6.2.3 Layer 3 Architecture

Layer3 has the Decision-making Agent (DA), Replication Agent (RA) and Profile Database (PRDB).

Decision-making Agent (DA)

Decision-making Agent (DA) is the highest entity and uses MA to control and coordinate with every host agent and net agent in this system. It commands the agent to execute specific functions such as updating file-rules dynamically. DA is the heart of the entire framework. DA is responsible for the analysis of the information collected by MAs. It has a global view of the system and has knowledge of: i) availability of resources in the hosts (memory, disk space, etc.). It has a knowledge of overloaded nodes and the details of which agent was received by the nodes. ii) Necessity of agents of the network, for example, whether they need more memory. It identifies when an agent is necessary, which indirectly indicates that its replication strategy should be observed and possibly altered; iii) which action of recovery must be executed for each type of detecting faults.
On the event of an agent failure, the decision making agent is capable of performing replication with a group of agents, migration or creation of new agents. If the user accesses unauthorized a software application which is not in this repository, then such accesses are also declared as intrusions. Similarly, the usage of the system resources by the user process exceeds the limit. The manager can take the decisions whether the particular user can access the software applications or not by modifying the profiles.

**Replication Agent (RA)**

Replication Agent (RA) is responsible for replication and recovery management. In other words, for the organization of the group of replicated agents, adding or removing an agent from a group; for the strategies of replication of each group and alteration is necessary; for the consistency of the replicas. The replication is transparent to the agents. Only RA knows the groups, the strategies and the number of replicas in each group. RA has knowledge of the localization of the active agents and its replicas.

**Profile Database (PRDB)**

PRDB is the database of profiles. It stores the information related to the standard behavior of each agent on the network. All the authorized actions of each agent in the system should be registered, besides the agents that can send solicitations to agents that are registered. DA uses this information in the process of evaluation for the detection of malicious agents.
Fault Tolerance

Fault tolerance can be a logical or a physical one. The decision-making agent analyses the data collected by MA and passes the control to replication agent. If it is a logical problem the RA corrects the network. If it is a physical problem it sends a SMS to the network administrator. When the user logs on the particular system, the user id and system details are communicated to the manager on the server side.

6.3 PROCEDURE OF INTRUSION DETECTION MECHANISM

Step 1: Take the backup for the entire system to avoid data loss. Search for the IP address in the Workgroup. Use Workgroup name for searching, directory entry with “WinNT” command.

Step 2: Activities done by system users, either authorized or unauthorized, can be monitored.

Step 3: Client has the IP address of the server which then monitor the files in the drive
if (intrusion detected)
{Restore the same file to the exact location
Send alert info to the server about the correction}

Step 4: Receive the info coming from the client machine and display to the server (Admin for verification). Information obtained from the intrusion detection system can be utilized to enhance the overall security of the network.

Step 5: Always have a connection with all the systems
if (ping result)
{Message as success}
else
{Display message as error}
6.3.1 Explanation of the Algorithm

When a process wants to find out whether the particular user is intrusive or not, it sends a CALL message to its agent profiles database (PRDB). The first parameter of the CALL message is the id of the user that propagates the message and the second parameter is the IP address of the initiator. When the decision making agent (DA) process receives the CALL message from initiator process, it performs one of the following actions.

The network agent identifies that the user is unauthorized.

- A client has entered into the network.
- The IP address is read in real-time of the network through a net agent on the destination machine.
- If the unauthorized user comes in the network then the net agent identifies the incoming user and concludes that the user is unauthorized. Hence the net agent decides to block the user. The unauthorized user information and IP address is updated in the database.
- A net agent detection engine is used to identify pre-defined patterns of misuse. If a pattern is detected; an alert is generated and forwarded to a central console or to another agent in the network. The intruder information is available in the profile database (PRDB).
- A system administrator is notified and the response is generated with the help of Decision Making Agent (DA).
- The alert is stored for later review and correlation.
- Reports are generated summarizing alert activity.
The network agent and host agent identify that the user is authorized.

- A client has entered into the network.
- The IP address is read in real-time of the network through a net agent on the destination machine.
- If the authorized user comes in the network then the net agent identifies the incoming user and extracts the given user’s information from the Profile Data Base (PRDB) and concludes the user is authorized. Hence the net agent decides to allow the user. The authorized user information and IP address is updated in the database.
- After the client has entered into the network, the host agent identifies whether the authorized user can misuse their rights or not. The event record may be created when an action occurs; such as a file is modified. The record is written into a file that is usually protected by the operating system trusted computing base.
- The host agent generates an event ID and transmits the user ID and the host IP address to the Mobile Agent (MA). This happens at predetermined time intervals over a secure connection.
- The Mobile Agent (MA) detection engine, configured to match patterns of misuse, processes the file. A log file is generated that becomes the data collection for all the unprocessed data that will be used in inspection.
• An alert is generated. When a predefined pattern is recognized, an alert is forwarded to a number of various agents for notification, response, and storage.

• The server administrator is notified.

• A response is generated. The response subsystem matches alerts to predefined responses or can take response commands from the security officer. Responses include reconfiguring the system, shutting down a target, logging off a user, or disabling an account.

• When an agent fails, the integrity of the data set has to be preserved. The server must have a way of reallocating a subtask to another agent. The replication is transparent to the agents.

• The alert is stored. The storage is usually in the form of a database. Some systems store statistical data as well as alerts.

6.4 DISTRIBUTED MOBILE AGENT RESOLUTION

A distributed system requires dynamic communication architecture, and is based on the client/server model. Heterogeneity can be expected from agents in the application. An agent designer can expect agents to interact with what is commonly referred to as middle or broker agents. Middle agents are responsible for what is often called yellow pages service, which means that the agents take requests from one agent, locate another agent that can fulfill the request, and put the two in contact. A complete treatment of security includes a way of making sure that an agent is actually the agent it is identified as, providing some data encryption policy so that unwanted agents cannot eavesdrop or intercept data being passed from one agent to another,
and making sure that only agents that have permission are allowed access to the system.

Distributed MAS (DMAS) technologies and their services show their inherent modularity and ease with which they can be recombined to form new applications. When designing new distributed software systems, however, the broad requirements and their translations into specific implementations are typically addressed by partial, complementary and overlapping technologies.

The goal of a mobile-agent system is to have a positive impact in any combination of the following three areas: (1) to augment a human end user's information-based perceptual capabilities by reducing information overload and providing context-relevant information, (2) to qualitatively and quantitatively improve the range of actions and activities which the end user can engage, and (3) to enhance the resources typically through the context aware use of devices, as is done in pervasive and ubiquitous computing, by which humans may perceive the world or by which humans may affect their decisions within it.

6.5 PERFORMANCE ANALYSIS

To test the validity of the improved system, this section designs an experiment to hold back the stimulated some kinds of attack activities with two different systems. One is the improved intrusion detection system based on Agent, and the other is a general Rule-based intrusion detection system. Finally, concludes by analyzing and comparing with the test result.
6.5.1 Experiment Presentation of the Algorithm

The experiment is carried out in a LAN with a bandwidth of 100 Mbps. C# is used as the language for writing the code for the improved IDS. The attack activities are simulated with the target as a host on the LAN. All types of attacks are performed on the proposed Intrusion Detection System (IDS) architecture. Let us now measure the DOS attack type of various nodes. The proposed architecture carries the higher detecting performance. In order to compare the result, the number of attack activities is controlled to be 1000, 1500, and 3000. Table 6.1 compares the performance of proposed architecture with all distributed intrusion detection procedures in the literature in terms of attack number, detected number, missing number and error number. With all four performance measures, Table 6.1 clearly shows that the performance of the proposed algorithm is better or equal to the existing algorithms.

Table 6.1 Performance Comparison of DIDS for Detecting DOS Attacks

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Attack Number</td>
<td>1000 1500 3000</td>
<td>1000 1500 3000</td>
<td>1000 1500 3000</td>
<td>1000 1500 3000</td>
</tr>
<tr>
<td>Detected Number</td>
<td>993 1490 2989</td>
<td>995 1492 2990</td>
<td>1000 1497 2995</td>
<td>1000 1498 2997</td>
</tr>
<tr>
<td>Missing Number</td>
<td>10 15 20</td>
<td>6 7 5</td>
<td>2 5 5</td>
<td>2 4 3</td>
</tr>
<tr>
<td>Error Number</td>
<td>3 5 9</td>
<td>2 4 7</td>
<td>2 2 0</td>
<td>2 1 0</td>
</tr>
</tbody>
</table>
6.5.2 Performance Analysis of the Algorithm

The comparison of DIDS algorithms for the number of detected and corrected attacks is shown in Table 6.2. The percentage of correction increases with the number of attacks. This is shown in Figure 6.2. Therefore, it is said with the experiment that the improved intrusion detection system based on agent gains higher detecting performance with fault tolerance.

Table 6.2 Comparison of DIDS Algorithms for the Number of Detected and Corrected Attacks

<table>
<thead>
<tr>
<th>Number of Attacks</th>
<th>Detected Number</th>
<th>Corrected Number</th>
<th>Error Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>494</td>
<td>492</td>
<td>2</td>
</tr>
<tr>
<td>1000</td>
<td>990</td>
<td>987</td>
<td>3</td>
</tr>
<tr>
<td>2000</td>
<td>1985</td>
<td>1979</td>
<td>6</td>
</tr>
</tbody>
</table>

This system is provided with high detectivity. By the introduction of Mobile Agents, the system is able to analyze and classify in real-time the state of intruding activity and hence it can detect the attack correctly.

Figure 6.2 Performance Analysis
This system is also provided with high flexibility and fault tolerance. To prevent allowing the whole networks go undefended. When a part of the IDS fails, agents can work autonomously even if their creators don’t operate anymore.

6.6 SUMMARY

In this section, a software agent is a software entity, which functions continuously and autonomously in a particular environment, often inhabited by other agents and processes. With a Mobile Agent (MA), agents act individually and/or in cooperation with other agents to fulfill goals set by their initiations and thereby maximize some expected utilities. In this work, a Mobile Agent (MA) for a pre-hit reconfiguration is developed. This Mobile Agent (MA) consists of agents that perform various tasks locally, and cooperate with each other to achieve a bigger goal.

The proposed procedure has been applied on several networks. The experimental results testify that the method proposed, can estimate intrusion detection mechanism faster and better. It also includes a comprehensive accuracy testing and can be performed with both host-based and network-based metrics. By using different sets of data and comparing different methods, the experimental results have demonstrated the effectiveness of the proposed approach. Good intrusion detection results have been obtained for these complex sequences. The proposed approach represents an improvement in the intrusion detection ability when compared to a well-known intrusion detection approach. During the experiment, it was observed that the improved cluster based dynamic distributed intrusion detection system with fault
tolerance architecture works on different kinds systems. This chapter proposed a new procedure that speeds up the intrusion detection mechanism, minimizes the time for detecting intruders, and reduces the bandwidth load, as compared with other centralized generalized intrusion detection procedures and DDIDS-MA.