Enhanced Intrusion Detection System with Mobile Agent

Trushna Tushar Khose Patil  
Department of Computer Engineering  
BSCOER, Pune  
Trushna.kaware@gmail.com

C.O. Banchhor  
Department of Information Technology  
SCOE, Pune  
cobanchhor@gmail.com

Abstract—This electronic The widespread proliferation of Internet connections has made current computer networks more vulnerable to intrusions than before. In network intrusions, there may be multiple computing nodes that are attacked by intruders. The evidences of intrusions have to be gathered from all such attacked nodes. An intruder may move between multiple nodes in the network to conceal the origin of attack, or misuse some compromised hosts to launch the attack on other nodes. To detect such intrusion activities spread over the whole network, we present a new intrusion detection system (IDS) called Distributed Intrusion Detection using Mobile Agents (DIDMA). DIDMA uses a set of software entities called mobile agents that can move from one node to another node within a network, and perform the task of aggregation and correlation of the intrusion related data that it receives from another set of software entities called the static agents. Mobile agents reduce network bandwidth usage by moving data analysis computation to the location of the intrusion data, support heterogeneous plat-forms, and offer a lot of flexibility in creating a distributed IDS. DIDMA utilizes the above-mentioned beneficial features offered by mobile agent technology and addresses some of the issues with centralized IDS models. The detailed architecture and implementation of a prototype of DIDMA are described. It has been tested using some well-known attacks and performances have been com-pared with centralized IDS models.

1. INTRODUCTION
An intrusion detection system (IDS) monitors network traffic and monitors for suspicious activity and alerts the system or network administrator. In some cases the IDS may also respond to anomalous or malicious traffic by taking action such as blocking the user or source IP address from accessing the network. IDS come in a variety of “flavors” and approach the goal of detecting suspicious traffic in different ways. There are network based (NIDS) and host based (HIDS) intrusion detection systems. There are IDS that detect based on looking for specific signatures of known threats- similar to the way antivirus software typically detects and protects against malware- and there are IDS that detect based on comparing traffic patterns against a baseline and looking for anomalies. There are IDS that simply monitor and alert and there are IDS that perform an action or actions in response to a detected threat. We’ll cover each of these briefly.

Issues with the existing centralized ID models

• Additions of new hosts cause the load on the centralized controller to increase significantly.
As a result, it makes the IDS non-scalable. Communication with the central component can overload parts of the network.

- Some of these IDSs contain platform specific components.

Distributed Intrusion Detection using Mobile Agents (DIDMA) that addresses some of the issues with centralized ID models as mentioned above. DIDMA uses two sets of software entities called static agents and mobile agents. Mobile agents are software components that can migrate to all the hosts in the network, and autonomously execute the tasks of detecting intrusions. Static agents are per host entities, and responsible for indicating all the hosts on which suspicious activity is detected.

Mobile agents visit all these hosts on which the static agents detect suspicious activities. The mobile agent at every visited host gathers the portion of the data that contains attack trace form the static agent running on that host and aggregates and correlates it with the data it has received from other static agents that have detected the same type of suspicious activity. The resulting data is carried over to the next host to be visited and the same process is repeated at every visited host. This technique results in decentralized data analysis carried out by mobile agents that make the IDS more scalable. Moreover, small sized mobile agent code and attack trace data carried by mobile agents may cause less load than sending large amount of raw data sent over the network.

I. DIDMA ARCHITECTURE

The presented intrusion detection system, DIDMA is designed by keeping in mind the notion of flexibility, scalability, platform independence, and reliability. The components of DIDMA are as follows: Static Agents (SA), Mobile Agents (MA), Mobile Agents Dispatcher (MAD), VHL, Alerting Agent (AA) and IDS console. The VHL is a subcomponent of the MAD. The architecture of DIDMA is shown in Fig 1.

An SA generates an ID event when suspicious activities are detected. This activity can be failed login attempts, suspicious connections, port scanning, or modification of system sensitive files from suspicious users. The SA sends ID events related to such behavior on the host to the MAD, which then creates an MA to handle the task of detecting intrusions based on such activities. The VHL contains lists that store the IP addresses of hosts on which suspicious activities are detected. The address of the SA that generated the ID event is added to a list in the VHL. An MA visits all the hosts listed in the VHL. An MA gathers traces of an attack form the SA running on a visited host, and aggregates and correlates the collected information with the data it has received from the other SAs that have generated the same type of ID event. The resulting data is carried over to the next host to be visited as guided by the VHL. The MA generates alerts and sends an alert to the console.
on the detection of any attack. An AA module receives generated alerts from MAs and displays the alerts to the security administrator using the IDS console.

Mobile agents are responsible for collecting evidences of an attack from all the attacked hosts and for further analysis of the gathered data. An MA takes the route specified in the VHL. Each MA contains code only for detecting a specific type of attack and hence less programming code and size of each MA. The IDS can be easily extended by adding new MAs for detecting new attacks, or the existing MAs can be modified for better detection capability, resulting in a highly modular and extendable architecture. The MAs take data from all the SAs running on hosts listed in the VHL, aggregate and correlate the data, and generate alerts.

The aggregation and correlation take place during its movement from one host to another, and any redundant data is removed. Since a single centralized module does not perform the aggregation and correlation of data collected from the SAs, it results in highly decentralized data analysis architecture. The failure of the MAD module does not interrupt the functioning of the MAs after it is dispatched. This allows the IDS to complete the currently running intrusion detection tasks even in case of failure of the MAD. For attacks like doorknob rattling, the data collected from SAs have to be just aggregated to detect the attacks. However, in chain or loop attack, the data have to be both aggregated and correlated. The alert generated by an MA is sent to alerting subsystem contained in the AA.

B. Static Agent (SA)

Static agents act like host monitors generating ID events whenever traces of an attack is detected, and these events are sent in the form of a message to remote object in the MAD. Each ID event carries information of the probable type of attack. For example, an SA identifies failed password guessing attempts as a suspicious activity, and an ID event is generated to check for doorknob-rattling attack. Again, a large number of connections generated from a host within a short period of time destined for a single target will trigger an ID event to indicate DoS activity on the host. An SA is a multithreaded program where each thread monitors the host for different classes of attacks. The SA is responsible for parsing the log files, checking for intrusion related data pattern in log files, separating data related to the attack from the rest of the data, and formatting the data as required by the MA.

C. MA Dispatcher (MAD)

The MA Dispatcher (MAD) decides which MA has to be dispatched according to the ID event generated by an SA. MA originates from this component. The MAD is a program that initiates an object request broker server. An SA communicates with the MAD using the proxy of the objects created at the MAD for sending event messages.

These objects are then responsible for creating an MA and sending it to the victim host(s). The MAD contains another component called Victim Host List (VHL) that maintains separate lists to store the IP addresses of all the hosts that are subjected to same types of attacks. For example, all the hosts subjected to doorknob rattling attack are maintained as a separate list in the VHL. The VHL provides the itinerary for the movement of an MA within the network. When the MAD receives an ID event message from an SA, the IP address of that SA host is added to the respective list in the VHL.

D. Alerting Agent (AA) and IDS Console

An alerting agent receives alerts generated from MAs. The AA sends these alerts to an IDS console that can be used by the security administrator to view alerts generated from the IDS.
The AA is also responsible for preventing multiple alerts being generated. The AA also stores the information gathered from an MA for further analysis.

II. MOBILE AGENT ALGORITHM

Given a mobile agent system, define a mobile agent algorithm. To each mobile agent is associated a transition system that can interact with the execution places and the navigation subsystem.

Let Qp be a (recursive) set of states associated to the execution places, and let QA be a (recursive) set of states associated to the mobile agents. The initial state of each mobile agent a is λ(a) and the initial state of each execution place p is λ(p).

Let p be a place. We denote by state(p) the state associated to p. Let a be an agent. We denote by state(a) the state associated to a.

The transition associated to the mobile agent a in the state s on the place p in the state q, transforms s into s′, q into q′ and either a does not move or it migrates on an adjacent place through the port out. We denote the transition by:

\[(s, q, \text{in} \mid \text{out}) \rightarrow_a (s', q', \text{out})\]

It means that the mobile agent a has migrated on the place p through the port in or after the transition it leaves the place p through the port out, with the convention that if the agent was already on the place and it does not move after the transition then in = 0 and out = 0; furthermore in and out cannot be simultaneously different from 0.

A configuration of the mobile agent system consists of the state of each place, the state of each agent, the collection M of agents in transit (initially M is empty) and a mapping π describing the placement of the agents which are not in a channel (several agents can be on the same place).

An event in the mobile agent system is defined by a transition associated to an agent a on a place p, it has the form:

\[(s, q, \text{in} \mid \text{out}) \rightarrow_a (s', q', \text{out})\]

In general, names are not available to the places themselves. Nevertheless, for ease of exposition, an agent a in transit is denoted by (p, a, p′) where p is the place of departure and p′ is the place of arrival. The state of each agent different from a is not affected, the state of each place different from p is not affected, the new state of a is s′ (it was s before the event), the new state of p is q′ (it was q before the event), and:

1) If in = 0 and out ≠ 0 then the set of agents in transit after the event is M ∪ {(p, a, p′)} (where p′ is the adjacent place of p corresponding to the port out,) and π is no more defined for a and unchanged for the other agents,
2) If in = 0 and out ≠ 0 that is (0,1) then 
3) If in ≠ 0 and out ≠ 0 that is (1,1) then process is completed and go to step 4.
4) Stop.
IV. MA FLOWCHART

Steps of flowchart:-
Where:-

- Initialization state \( s \) for \( \lambda(a) \) and initialization state \( q \) for \( \lambda(p) \).

- After the migration \( s \) become \( s' \) and \( q \) become \( q' \).
- Conditions .

V. CONCLUSION

We have presented a distributed intrusion detection system using mobile agents called DIDMA, which addresses some of the disadvantages of the centralized distributed intrusion detection systems.

DIDMA employs static agents as host monitors and mobile agents for data collection, aggregation and correlation, and to respond to any attack. DIDMA exploits the benefits of employing mobile agents such as reduced network bandwidth usage, increased scalability and flexibility, and ability to operate in heterogeneous environments.

DIDMA offers a new technique for decentralized data analysis carried out by mobile agents at the site of audit data instead of sending the audit data to some central data analysis component.

VI. REFERENCES


