International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2018 IJSRCSEIT | Volume 4 | Issue 3 | ISSN : 2456-3307

Distributed Intrusion Detection Using Mobile Agents

F. Asmathunnissa[1], A.Lavanya[2]
[1]M.Phil (Research scholar), Kamban College of Arts And Science For Women, Thiruvannmalai, Tamil Nadu, India

ABSTRACT

DIDMA (Distributed Intrusion Detection using Mobile Agents) is a novel architecture in the field of IDS (Intrusion Detection Systems), utilizing an agent-based approach in order to realize a distributed framework. The novelty in this architecture is the employment of mobile agents as its auditing components. This novel approach overcomes certain problems associated with traditional designs in IDS. In particular, problematic areas such as high-speed networks, not visible traffic, and fail-open architecture have been successfully managed. Moreover, the fault tolerant decentralized design of DIDMA clearly demonstrated resilience against active attacks.

Keywords: Intrusion Detection, Network Security, Distributed Networking, Mobile Agents

I. INTRODUCTION

The security of a local networking environment, or a single computer system, is a very important aspect of its infrastructure. A properly secured system is able to guarantee confidentiality, integrity, and availability of its resources and services. Thus, the ability to protect a system from outside interference is of the highest importance. However, individuals can specifically attack flaws in computer systems. These attacks result in rendering the system vulnerable, and compromising its entire security scheme. One aspect of the general security scheme is to detect if and when an attack against protected resources is attempted. The topic of network security that deals with this field is IDS [1]. The intrusion detection technology, employed in order to monitor resources distributed among several nodes in a local network, adopts a distributed approach to its design. However, contemporary IDS still suffer from many problems, like the inability to handle large amounts of network traffic, not visible network traffic, and a central monitor that provides a single point of failure, among others. The DIDMA prototype proposes a novel architecture that approaches the aforementioned problems by employing a distributed solution using mobile agents [2]. The components of the system are realized as software agents that have certain properties, and provide several advantages over traditional intrusion detection approaches. These software agents are its auditor components, and are implemented by using the Grasshopper platform [2]. Handling the information collected by an intrusion detection system requires extensive cross referencing in order to identify distributed attacks. Therefore, there is an obvious need for an automated distributed approach to the design procedure of detecting security violations. The rest of this paper is organized as follows. The second section presents the current framework of IDS and the associated problems, along with a brief introduction to the mobile agents technology. Moreover, related work is presented in the second section. The third section analyzes the DIDMA architecture. The results of the proposed design and possible future work are presented in the conclusion.

II. CURRENT FRAMEWORK

A. IDS and Existing Problems

Intrusion detection is a type of network security that aims to detect, identify, and isolate attempts of intrusion, or unauthorized usage of computer and network resources [1]. There are some problems that infest the traditional design approaches. IDS are not able to reliably intercept and examine the *high-speed traffic* of contemporary networking environments. The speed of networks increases at such a rate that IDS require many resources and computational power. Although the development of the IDS technology will continue to improve, likewise the speed of networks will continue to grow. Another problem closely related to the previous one is *not visible traffic*. The currently used high speed networks employ switched topologies in order to increase their speed. Although this approach has many benefits for the performance of the network, it interferes with the operation of IDS, because it distributes the traffic in ways that make it invisible to certain parts of the network. Furthermore, the employment of Virtual Private Networks (VPN) that encrypt the exchanged network traffic also creates monitoring problems for IDS. Consequently, a system deployed on such networks is unable to perform correctly. Nearly every intrusion detection system has the problem of *fail-open architecture*. If an attacker finds a way to disable the system, usually through the use of denial of service type attacks, then the previously monitored network becomes completely unprotected. Moreover, the system that failed provides absolutely no notification about its inability to continue providing detection services [3]. The *inability to detect certain types of attacks* is another common problem with IDS. Detection approaches are based on fixed, constant models that are matched against known attack patterns. Research on the field of detection approaches has proved that attacks which achieve the same results, but are slightly different than the ones expected by the system, are not detected [3]. Finally, one of the most important problems of IDS is their *high-rate susceptibility to false positives*. A false positive can be defined as the situation of threats that appear to be real to the system, but in reality they are just normal data transactions. This vulnerability of IDS is responsible for many problems, as an attacker can create many false positives, and when the system is reconfigured to ignore such patterns, he can launch the real attack undetected. Moreover, if the system is configured to respond aggressively to detected attacks, then a reaction to a false positive situation results to interference with legitimate operations.

B. Mobile Agents Technology

Agents are defined as software programs that are situated in an execution environment and are characterized by the property of autonomy [2]. The property of mobility constitutes one of the most useful characteristics of agents, though not necessarily of intelligent ones, since it can prove to be advantageous in the networked environments that exist today. Moreover, mobile agents (MA) offer a cleanly designed solution to the problem of distribution of a specific task over several entities that work together to achieve a common objective. The mobile agents' technology provides a development environment that can be utilized by designers in order to produce robust distributed applications that are able to operate efficiently in the contemporary wide area networks that are characterized by their dynamic nature. An investigation of the mobile agents technology reveals many advantages over the established distributed networking models.

C. Related Work

The problem of designing an IDS that follows the distributed networking paradigm has been researched in the past, and several prototype systems have been implemented. However, the technology of MA has not been very widely utilized in the intrusion detection field, and generally in the field of network security. The reason is basically the fact that this is a rather new approach to distributed computing and has not yet found many applications. The Autonomous Agents for Intrusion Detection (AAFID) project has approached the problem of intrusion detection with an agent-based solution. The project defines autonomous agents as independent software entities that are responsible for a specific monitoring task at a single host [5]. The proposed design of the system follows the distributed paradigm since it is based on multiple independent entities that work collectively and share information. The collected findings are reported to a central monitor that processes the data and is able to detect intrusions and intrusion attempts [5]. Although the system utilizes a distributed philosophy and design, it does not employ mobile agents. The agents of the system do not have the ability to migrate to other systems, but are stationed to a single host. Furthermore, the central monitor is a single point of failure, and can be attacked by a malicious party in order to render the system unusable. Another project that aimed to decentralize intrusion detection is the Distributed Intrusion Detection System (DIDS). The system employed several sensors in a restricted networking environment that monitored host and traffic events. Although the system distributed the monitoring to several different places in the network, there existed a single centralized director that received information from the sensors and reported intrusions [6]. Moreover, the developed system employed no agent-based technology or methodology. Thus, it was unable to receive several benefits of such an approach, like a true decentralized design that can robustly withstand an attack, and successfully overcome the problem of fail-open architecture.

A distributed administration system, named Sun Enterprise Network Security Service (SENSS), was recently released by Sun Microsystems. The system performs distributed administration functions in a local network environment by employing Java-based mobile agents. These agents are launched from a central location and they migrate to hosts in the network performing predefined administrative tasks [7]. It must be noted that the system was not developed to perform intrusion detection functions, thus it has certain limitations when it is employed to perform security-auditing procedures. The mobile agents it utilizes have no autonomy and they are able to travel only to a single host, not having the ability to migrate again. This fact proves to be a shortcoming since non-autonomous agents depend on the parent process, thus their functionality is reduced in the case of an unanticipated problem. The design process of the DIDMA system proposes a framework upon which further functionality can be added and implemented.

III. SYSTEM DESIGN

This approach has the benefit of abstracting and breaking down the problem into several parts that can be managed with greater ease. Thus, a simple blueprint of the design of the system can prove to be very helpful in expanding its basic functionality. Figure 1: Simple representation of the DIDMA system the system, at its most basic level, defines two different kinds of agents. The first are stationary agents that are situated at specific hosts in the local area network that must be monitored for intrusion attempts. The system refers to these entities as sensor agents (SA). The SA are responsible for collecting information about the specific host on which they are stationed. The data are collected from the log files of the system, as well as from operational statistics and events directly from the system. The collected data are processed by the SA and rendered into a common predefined format. It must be noted that the SA are not responsible for classifying events as intrusion attempts, nor are they able to alert the system of such an event. The party responsible for these activities is the second kind of agent defined by the architecture. The second kind of agents that exist in the proposed system are multi-hop mobile agents. These agents have the ability to transport themselves between the hosts that are under the surveillance of the intrusion detection system. They are responsible for inspecting and evaluating the information collected by the sensor agents. Depending on their predefined sensitivity, they can classify whether an event has been an intrusion attempt. When a MA classifies an event as an attack, it has the ability to alert the system. The proposed system is also consisting of a configuration console that provides a graphical user interface in order to interact with the party that has the responsibility of administrating the monitored network. The auditor agent configuration console (AAconf) provides the functionality of configuring certain run-time variables of the auditor agents and launching them to the network environment.

A. System Components

a. Sensor Agents

The SA is the sensor component of the developed intrusion detection system. The SA is a stationary agent that is bound to a specific host that is under the surveillance of the intrusion detection system. The SA is accompanied by a file, the attack patterns signature file that contains patterns of known attacks. The SA against the log files of the hosting system periodically checks these signatures. Any matching entries in the log files are extracted by the SA, formatted according to predefined format, and placed in the audit file. Therefore, the SA must be aware of the log files of the hosting system and the format utilized by those. The SA is the only component of the system that is platform dependent. If a new operating system needs to be supported and the corresponding host placed under the monitoring of the system, a SA that specifically understands the log files generated by this operating system should be designed. Moreover, this SA should be accompanied by an attack pattern signature file that contains known attacks related to the hosting operating system. Another detection approach that can be accommodated by the sensor agent is the incorporation of legacy, non-agent oriented, intrusion detection software [8]. A network-based intrusion detection system, like Snort [9], can be wrapped by an agent interface and monitor network activity for known attack signatures. The SA wrapper performs the translation between external agent requests to the legacy code, and between the legacy system's requests to the agent communication protocol [8].

b. Auditor Agents

The auditor agent subsystem is in essence the auditor component of DIDMA. The AA component is realized through a mobile agent that visits each host that is protected by the system and checks whether there has been a security violation that needs to be reported. The mobile AA knows from the moment that is launched the list of Internet Protocol (IP) addresses that correspond to the hosts that must be audited. The choice of the host that is visited next is not random, but the agent implements the oldest-node algorithm, that specifies that the agent should visit the host it last visited longest ago (or never visited, or does not remember visiting)[10]. When the mobile auditor agent visits a host, it accesses the audit file that has stored the events that should be evaluated. The agent evaluates these events, and according to its specified sensitivity it generates an alert or not. The alert methods that the AA supports are three. A generated alert can be sent via an e-mail message by utilizing the predefined destination e-mail address and a Simple Mail Transfer Protocol (SMTP) host. Another way of alerting is via a Short Message Service (SMS) text that is sent to a predefined mobile terminal number through an SMS center number. Finally, the AA can alert by sending a Transmission Control Protocol (TCP) message, which contains the text of the alert, to a predefined host. The format of the message follows the Intrusion Alert Protocol (IAP) which is an application level protocol for exchanging intrusion alert information. The protocol provides transport and security characteristics that are required in order to exchange alert data over insecure channels [11].

c. Auditor Agent Configuration Console

The auditor agent configuration console component is the subsystem that is mainly responsible for the interaction with the user, and the configuration, creation, and launching of the mobile auditor agents. It can be installed at any host of the internal local area network, and after the creation of the mobile auditor agents it can be safely ignored, even completely uninstalled. The user installs this component, and through the provided functionality the mobile auditor agents are configured. After the definition of the configuration parameters, the mobile agents are created and launched in the local area network. Since the AA conf subsystem plays no other role, nor it implements any other required operations, the system has absolutely no dependence to it. Therefore, the AA conf subsystem can be deactivated, or even uninstalled, without interrupting the operation of DIDMA. The absence of a central management component confers a distributed nature to the developed intrusion detection system. Thus, the system has no single point of failure which could be attacked by an outsider and disable it.

B. The Audit File

The communication between the stationary sensor agent of a specific monitored host and the visiting mobile auditor agent is realized through the audit file. The file should follow a predefined format in order achieve interoperability with other systems. Since the developed intrusion detection system employs agent-oriented software engineering approaches, the format of the audit file should be compliant to both intrusion detection and software agent standards. This fact guarantees interoperability with other intrusion detection systems that could be employed in a local area network, and also with other agents that belong to other development platforms and could utilize the resources of the present intrusion detection system in order to satisfy their own design requirements. Therefore, it gains all the advantages of the XML meta-language, like filtering and aggregation, and becomes easily extensible in order to be utilized by third party applications. Furthermore, the Knowledge Query and Manipulation Language (KQML) defines a standardized language for exchanging information and knowledge between agents. The main focus of KQML is to describe an extensible set of performatives that can be utilized in order to define the permissible operations that agents may attempt on the knowledge stores of each other [13]. Moreover, KQML provides a basic design architecture for knowledge sharing through agents. These communication facilitators coordinate the interaction and the exchange of knowledge and information between software agents [13]. The proposed format of the audit file follows a format that embeds the IDMEF in the KQML general directives. This is possible since KQML places no restriction in regard to the language of its messages, therefore IDMEF can be utilized.

C. System Implementation

A prototype of the DIDMA architecture has been implemented by utilizing the Java programming language, and the Grasshopper mobile agent platform [2, 14]. The Java programming language has been selected as the implementation language for its platform independency, and its strong security framework. Security features of Java, such as the automatic bounds checking, and the code verification signing, form a safety net for critical applications.

The Grasshopper mobile agent platform was chosen since it is readily and freely available for non-commercial endeavours. Grasshopper provides multi-protocol support, a naming service for migrating agents, advanced security features based on strong cryptographic algorithms, and a rich, fully documented interface for the creation of mobile, or other, software agents.

IV. EVALUATION

The approach of employing a distributed design based on MA for the infrastructure of an intrusion detection system, offers certain advantages over the traditional monolithic systems. These advantages form the motivation of DIDMA, the proposed architecture. The most important of these benefits addresses the problem of network traffic that is not visible to a traditional intrusion detection system, due to switched network environments. However, an intrusion detection system that employs the proposed design for the auditing process is able to monitor every host in the local network, despite the existing topology, since the mobile agents are free to travel to all nodes. Mobile agents can successfully resolve the inability of IDS to monitor high-speed networks. This central point of control is in essence a single point of failure, and constitutes the most probable target for an outside attacker. A distributed approach to the design of an intrusion detection system avoids this threat, since there are many mobile agents in the network that can alert interested parties if an attack on a node is detected. Although the proposed system defines a configuration management console, this is strictly for demonstration purposes. After launching the agents in the network, the configuration console can be completely disabled and the system will continue to operate normally. Furthermore, the proposed architecture can overcome the problem of false positives with its modular design. The mobile agents perform the auditing process on every host of the network and are able to cross-reference their findings. Thus, distributed attack patterns can be detected, and legal transactions can be confirmed. Moreover, the auditor agents can be configured to certain degrees of sensitivity dynamically, therefore the system is able to reduce the rate of false positives even further.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented DIDMA, a novel distributed architecture, upon which a prototype has been implemented. This implementation offers certain benefits to the intrusion detection methodology, and complements the traditional IDS technology. The detailed investigation that has been performed confirmed advantages of DIDMA over traditional IDS design approaches. The developed prototype was evaluated based on operational criteria, like the response time between an attempted intrusion and the actual alert, and the computational load on the hosting resources. Moreover, the fault tolerant architecture of DIDMA proved to be beneficial on active attacks against the system itself. The DIDMA architecture does not yet address the aspect of communication between the auditor mobile agents. Future work will be focused on the communication mechanisms between the auditor agents, in order to be able to recognize attack trails that spread over many monitored systems by cross-referencing their findings. Another area of possible further work is the realization of a minimal mobile agent platform that provides just the functionality and security mechanisms required by the DIDMA system. The advantages of implementing a specialized platform would be the low overhead of the system, and a distributed framework under the complete control of the intrusion detection system.

VI. REFERENCES

[1] J. McHugh, A. Christie, J. Allen, Defending Yourself: The Role of Intrusion Detection Systems, IEEE Software Magazine, Vol.17, No.5, 2000.

[2] W. R. Cockayne, M. Zyda, Mobile Agents, Prentice Hall, 1998.

[3] S. Northcutt, Network Intrusion Detection: An Analysts' Handbook, Second Edition, New Riders Publishing, 1999.

[4] J. J. Ordille, When Agents Roam, Who Can You Trust?, Bell Labs Computing Science Research Center

[5] J.S Balasubramaniyan, J. O. Garcia-Fernandez,D. Isacoff, E. Spafford, D. Zamboni, An Architecture for Intrusion Detection usingAutonomous Agents, CERIAS Technical Report 98/05, Purdue University, 1998.

[6] B. Mukherjee, T. L. Heberlein, K. N. Levitt, Network Intrusion Detection, IEEE Network Magazine, Vol.8, No.3, 1994.

[7] Sun Microsystems Inc., Sun EnterpriseNetwork Security Service