SAPA: Software Agents for Prevention and Auditing of Security Faults in Networked Systems

Rui Costa Cardoso and Mário Marques Freire

Department of Informatics, University of Beira Interior,
Rua Marquês d'Ávila e Bolama,
P-6201-001 Covilhã, Portugal
{rcardoso, mario}@di.ubi.pt

Abstract. This paper describes the design and implementation of a multi-agent system to detect and audit host security vulnerabilities. The system uses agent platforms allocated through the network to scan and interact with each host. The information collected by each agent is then used to build a common knowledge base that together with data retrieved from vulnerabilities information sources is used to improve the overall security. This approach reduces the total time to scan the network and the processing time overhead associated. The amount of traffic involved is also reduced. It allows the dissemination of updated knowledge about the network security status and reduces the communication with the network administrator. This solution provides an autonomous and proactive distributed system. It acts as a vulnerability assessment tool to make security notifications only if needed.

1 Introduction

In order to guarantee a global security solution in a given enterprise network it is necessary to take into account several issues such as: security mechanisms for exchange and access to remote information, mechanisms for protection of networked systems and administrative domains, detection of new vulnerabilities and exposures, monitoring and periodic audit of the implemented security mechanisms, and disaster recovery plans. This paper is focused on the problem of detection of security vulnerabilities in a proactive way, using software agents. Network security risks are rising continuously [1,2]. As networks become more interconnected, the number of entry points increases and therefore exposes each network to threats. The widespread of Internet access allows the dissemination of new vulnerabilities and hackers know-how. The Networks and applications are becoming more complex and difficult to manage and there is not a significantly increase in the number o human resources allocated. Also software development lifecycle is shorter resulting in flawed or poorly tested releases. Hackers have better tools, which require less technical skills and allow large scale attacks. The time between the identification of new vulnerabilities and the exploit attempt

C. Kim (Ed.): ICOIN 2005, LNCS 3391, pp. 80–88, 2005.

has been substantially reduced, giving less time to administrators to patch the vulnerabilities. Moreover, hackers often access to that information before the vendors are able to correct the vulnerabilities and it is difficult for network administrator to keep update with the patches. On the other hand, detection of security faults (holes) in hosts can anticipate the occurrence of service failures and compromises.

There are several approaches to the problem of vulnerability assessment. NIST (National Institute of Standards and Technology) [3] gives a good overview of the subject. Basically, there are open source approaches based on software packages like Nessus [4] and SARA [5] and commercial approaches by SecurityMetrics [6], Qualys [7] and ICSA [8]. The approach of security companies is mainly concentrated on the development of automated security programs capable to analyze the attacks within a single system such as Nessus [4], Nmap [9], SAINT [10], SARA [5] and SNORT [11]. All of these software products use a standalone approach and never share knowledge except when downloading updates from the central server. They examine system and network configurations for vulnerabilities and exploits that unauthorized users could use. Usually they act from a specific host in the network scanning all the others. Although most of them are programs and scripts run periodically by network administrators, its use lead to a rise in consuming time in which they are installed and also in the bandwidth availability, other negative remark is that this procedures may eventually lead to overhead in the performance of systems which could cause instability and crash in the scanned systems. Because there is not any exchange of data between applications there is not any guaranty that all share the same knowledge. On the other hand, some research work has been reported using software agents for network vulnerability scanning [12,13]. It uses a mobile agent approach and therefore can, if implemented correctly reduce the overall communication traffic in the network. Because mobile agents are not always in contact with its origin host, they have a reduced interaction when doing some task. They also allow the user to create specialized services by tailoring the agent to a specific task. The drawback of this approach is the security issues that arise when using mobile agents in an insecure network; this could lead to eventual content alteration. Is the agent trustable? And its content is authenticated? An interesting approach to the security problem in FIPA Agents platforms was made by Min Zhang et al. [14], which could be used to solve some of the mobile agents problems previously refereed. This paper addresses the problem of using a secure FIPA Agents platform for vulnerability analysis.

In this paper we present the design and implementation of an agent-based system, built using JADE [15], in which agents main task is detecting vulnerabilities and exposures [16]. Each agent can exchange knowledge with others in order to determine if certain suspicious situations are actually part of an attack, this procedure allow them to warn each other about possible threats. For external source of vulnerabilities [17] used to keep update the agent system we consider to use the ICAT Metabase [18], a search index of vulnerabilities in computerized systems. The ICAT binds the users the diverse public databases of vulnerabilities as well as patch sites, thus allowing us to find and to repair the existing vulnerabilities in a given system [19]. ICAT is not properly a database of vulnerabilities, but an index pointing to some reports of vulnerabilities as well as the information about patches currently available.

The remainder of this paper is organized as follows: Section 2 describes SAPA Agents. Section 3 discusses the distributed architecture. Section 4 is devoted to FIPA-ACL interactions. The Knowledge building is addressed in section 5. Main conclusions are presented in Section 6.

2 SAPA Agents

As more and more computers are connected to the network the security risk increases accordingly [20]. Although vulnerability assessment tools [21,22] and intrusion detection systems [23] are good solutions, they lack several features that we consider important, according [24]. Network Administrators needed to be permanently updated, and in control of everything that appended in their network, but it is widely known that this is not feasible in real time. An administrator has many routine and time-consuming tasks that could be delegated with advantages. Our solution is based in the delegation and cooperation; by delegating tasks to specific applications capable of autonomous behavior we can enhance the overall performance of the security in a network. In figure 1 we present a sketch of the platform. Our approach to the problem used JADE, a Java-based agent development framework [15,25], to evaluated the feasibility of the system. Jade is a FIPA-Compliant Agent Platform [26] in which we can deploy Java agents. The platform includes an AMS (Agent Management System), a DF (Directory Facilitator) and a sniffer RMA (Remote Management Agent). The features available includes: FIPA interaction protocols, automatic registration of agents with MAS, FIPA-compliant naming service and FIPA-compliant IIOP (Internet Inter-Orb Protocol) to connect to different APs (Agent Platforms). A GUI is also available to manage several agents and APs. SAPA Agents, acts as wrappers of VA (Vulnerability Assessment tools). JADE provide the infrastructure where the agents coexist and interact.

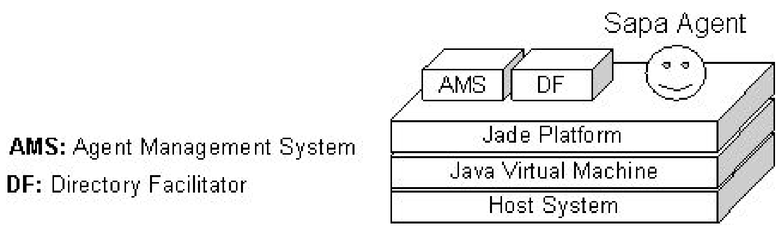

Fig. 1. SAPA Agent in a JADE Platform

2.1 Architecture

The SAPA agent has three main modules: DENOFAS module, FIPA-ACL Module, and Knowledge Module. The DENOFAS module [16] is used by SAPA to interact with the network. His main task is gathering valuable data to be processed. The FIPA-ACL [27] module used to exchange knowledge with other agents, and the Knowledge module used to maintain updated information about which Agents are active. In figure 2 we present the SAPA modules

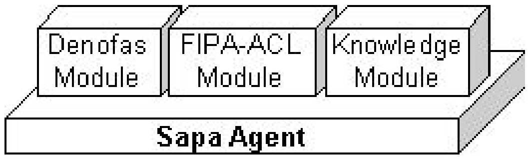

Fig. 2. SAPA Components

2.2 Denofas Module

The DENOFAS module acts has a vulnerability assessment tool and is used to scan the ports and services (not only in a specific host, but also the specific network allocated to him). All the data gathered is used by in the knowledge building. Each SAPA agent can by its initiative access the external vulnerability database [18] to extract new vulnerability data. Figure 3 represents the interactive process between SAPA agent and a Host.

2.3 FIPA-ACL Module

This module is used to interact with other agents. It allows the exchange of FIPA-ACL messages [27,29], between agents. The types of messages send are requests to fulfill several actions. The requests can be: Ask to acquire the knowledge of a specific SAPA Agent, disseminate warnings of possible attack based in new vulnerability posted in the database, inform about new updates in ICAT database, propagate vulnerability detection information, among others. Figure 4 represents the full FIPA-ACL compliant per formatives used in the exchange of FIPA Compliant ACL messages.

2.4 Knowledge Module

A brief overview of the Knowledge module follows. The data used to build the knowledge is supplied by the Denofas and by direct interaction with others agents through the FIPA-ACL module. The knowledge base allows the agent to act autonomously or in cooperation with others in the persecution of is tasks.

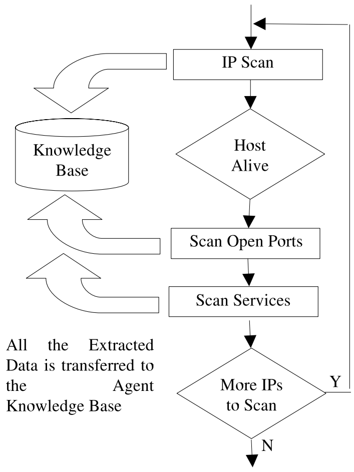

Fig. 3. DENOFAS Vulnerability Assessment procedures

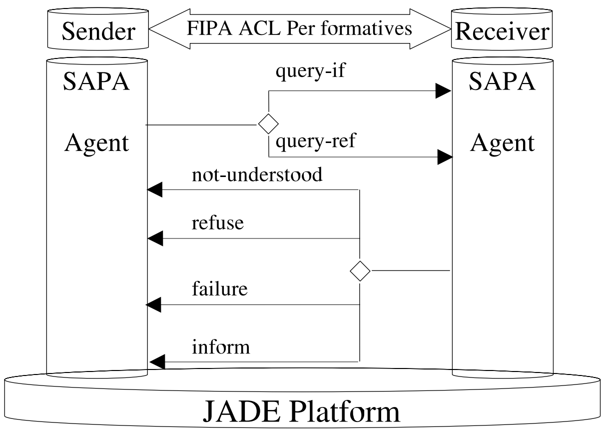

Fig. 4. FIPA-ACL Performatives

3 Distributed Architecture

In Figure 5 we present the overall interconnection solution based in distributed SAPA Agents (Software Agents for Prevention and Auditoring of Security Faults in Networked Systems). The SAPA Agent, receive solicitations to perform a specific task. The tasks currently supported are: host scan, network scan and host/network monitoring, in which the agent perform a detailed scan of the network, the open ports and the services active in those ports. After it collects all the requested data, it will use data from previous scans and data stored from several external sources to build is knowledge base. These sources are the ICAT [18] database of known vulnerabilities and exposures, the PortsDB [28] that tell us which services are associated with specific ports. After the inference process is complete and the agent has gathered sufficient knowledge about the situation, it will create an output result. Finally, presenting the output to the requester ends the process. The following Figure presents a summarized view of the Multi-agent System based in SAPA Agent interactions.

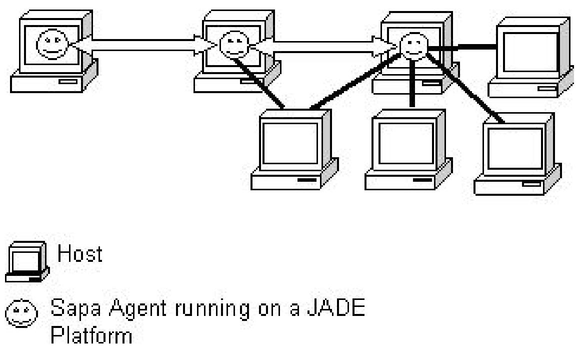

Fig. 5. SAPA MAS Multi-Agent System

The system is deployed in JADE [15] platforms based in hosts dispersed in the network. Each SAPA agent then can be given a task to monitor a specific host or group of hosts. The agents can also share information among them using FIPA-ACL messages [27]. The SAPA Agent main goal is to help network/host administrators to improve the security of computer systems networks. As already it was seen, the security problem arises, not only at the host level, where the agent is used to improve the security of a particular host, but also at network level. Being able to be used by network administrators, they could improve the overall security assessment of the network administrators. SAPA agents present a detailed assessment of the weak points in a network; making possible future security solutions to implemented. In this section we presented the architecture of our system.

4 Agent Interactions

In our multi-agent system we used FIPA-ACL messages to exchange information between Agents. Each Agent could: notify others that a specific host was listening in a suspect port, that ICAT data has been recently updated and also share is part of knowledge about the networks. Following we present an example of message exchange in our MAS using FIPA ACL (Agent Communication Language). The following messages represent a request/answer communication:

FIPA ACL Request:

```
(request :sender(agent-identifier:agentsapa1)
:receiver(agent-identifier :agentsapa2)
:content (action (agent-identifier:agentsapa2) open(data.txt))
:language fipa-sl)
```

The sender agent agentsapa1 send a request to the receiver agent agentsapa2, to update his data, using FIPA Semantic Language

FIPA ACL Failure:

```
(:performative failure :sender(agent-identifier:agentsapa2)
:receiver (set (agent-identifier :agentsapa1))
:content ((action (agent-identifier :agentsapa2) (open data.txt)
 (error-message No such file:data.txt))
:language fipa-sl)
```

The receiver agent agentsapa2 send a failure response to the sender agentsapa1 using FIPA-SL

5 Knowledge Building

The SAPA Knowledge Building process, integrate several data sources: online resources, information from others SAPA agents and the scanned data. ICAT [19] data used by the agent is automatic updated from the web. The agent uses the complete ICAT [18] database (exporticat.htm) and the information about all vendors, products, and version numbers contained within ICAT (vpv.htm). All this data is processed internally and is used as input to the Knowledge Base. In the knowledge building the agent also use the PortsDB [28] to specify what service is using what port. All this information is processed internally by the knowledge base. When the agent starts another scan it will use the data available to perform is task. Figure 6 illustrate the components of the knowledge building of the SAPA Agent.

6 Conclusions and Future Work

With this approach we achieve several goals: Reduce the response time, Propagate information more effectively. Free network administration from time consuming and routine tasks. Reduce substantially the burden of processing power in the overall by distributing it by several AP (Agent Platforms). The exploit of the same vulnerability in different systems became more difficult, because the agents help to detect it on time.

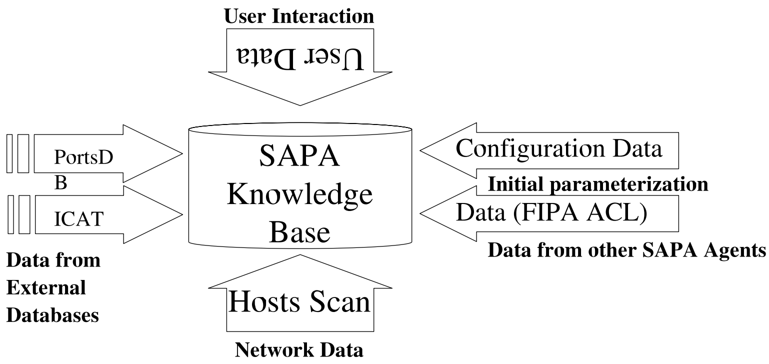

Fig. 6. SAPA Knowledge Building

7 Acknowledgements

The authors would like to thank the Jade Developer Community for useful discussions and precious suggestions. Part of this work has been supported by the Group of Networks and Multimedia of the Institute of Telecommunications - Covilh Lab, Portugal, within SAPA (Software Agents for Prevention and Auditing of Security Faults in Networked Systems) Project and by the EU IST EuroNGI (Design and Engineering of the Next Generation Internet) Network of Excellence, Sixth Framework Program, Priority 2, Information Society Technologies.

References

1. A. Householder, K. Houle, and C. Dougherty, Computer Attack Trends Challenge Internet Security, IEEE Computer, Security and Privacy - Supplement, April 2002, pp. 5-7.
2. CERT, http://www.cert.org. Accessed: 05/30/2004.
3. NIST: National Institute of Standards and Technology http://www.nist.org. Accessed: 05/30/2004.
4. Nessus, http://www.nessus.org. Accessed: 05/30/2004.
5. SARA: The Security Auditors Research Assistant http://www-arc.com/sara/. Accessed: 05/30/2004.
6. Securitymetrics. Integrated Vulnerability Assessment, Intrusion Detection and Prevention. Technical White Paper, Securitymetrics, 2003.
7. Qualys. On-Demand Security Audits and Vulnerability Management: A Proactive Approach to Network Security. Technical White Paper, Qualys. 2003.
8. Bace, R. , An Introduction to Intrusion Detection & Assessment. Technical White Paper, ICSA, 1999.
9. Nmap http://www.nmap.org. Accessed: 05/30/2004.
10. Saint http://www.saintcorporation.com. Accessed: 05/30/2004.
11. Snort: Open source network intrusion detection system http://snort.org. Accessed: 05/30/2004.

12. Taraka Pedireddy, Jos M. Vidal, A Prototype Multiagent Network Security System, Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems AAMAS03, July 14-18, 2003, Melbourne, Australia.
13. Jeffrey W. Humphries and Udo W. Pooch ,Secure Mobile Agents for Network Vulnerability Scanning, Proceedings of the 2000 IEEE Workshop on Information Assurance and Security, 6-7 June 2000, New York, United States, pp. 19-25.
14. Min Zhang, Ahmed Karmouch, Roger Impey, Adding Security Features to FIPA Agent Platforms.
15. JADE (Java Agent DEvelopment Framework) http://jade.tilab.com.
16. Rui Costa Cardoso and Mário M. Freire. An Agent-based Approach for Detection of Security Vulnerabilities in Networked Systems. In Proceedings of 11th International Conference on Software, Telecommunications and Computer Networks (SoftCom2003), Split, Dubrovnik (Croatia), Venice, Ancona (Italy), October, 7-10, 2003, pp. 49-53.
17. CVE: Common Vulnerabilities and Exposures http://www.cve.mitre.org. Accessed: 05/30/2004.
18. ICAT: Internet Categorization of Attacks Toolkit http://icat.nist.gov. Accessed: 05/30/2004.
19. Peter Mell, Understanding the World of your Enemy with I-CAT (Internet-Categorization of Attacks Toolkit), in 22nd National Information System Security Conference, October 1999.
20. J. P. Anderson, Computer Security Threat Monitoring and Surveillance, James P. Anderson, Co. FortWashington, PA, 1980.
21. Robert A. Martin, Managing Vulnerabilities in Networked Systems, in IEEE Computer, November 2001 (Vol. 34, No. 11), pp. 32-38.
22. R. A. Kemmerer and G. Vigna, Intrusion Detection: A Brief History and Overview, IEEE Computer, Security and Privacy - Supplement, April 2002, pp. 27-29.
23. C. Manikopoulos and S. Papavassiliou, Network Intrusion and Fault Detection: A Statistical Anomaly Approach, IEEE Communications Magazine, Vol. 40, No. 10, pp. 76-82.
24. B. Kim, J.Jang, and T. M. Chung, Design of Network Security Control Systems for Cooperative Intrusion Detection, in Information Networking, I.Chong (Ed.), Heidelberg, Springer Verlag, LNCS 2344, 2002, pp. 389-398.
25. F. Bellifemine at al., JADE - A FIPA-compliant agent framework, Proceedings of PAAM99, London, April 1999, pg.97-108.
26. FIPA http://www.fipa.org. Accessed: 05/30/2004.
27. FIPA ACL Message Structure Specification, http://www.fipa.org/specs/fipa00061/. Accessed: 05/30/2004.
28. PortsDB (Ports Database) http://www.portsdb.org. Accessed: 05/30/2004.
29. Y. Labrou, T. Finin, and Y. Peng. Agent communication languages: The current landscape, IEEE Intelligent Systems, March/April 1999, pp. 45-52.