A Policy Propagation Model Using Mobile Agents in Large-Scale Distributed Network Environments

Tae-Kyung Kim[1], Dong-Young Lee[1], Ok-Hwan Byeon[2], and T.M. Chung[1]

[1]Internet Management Technology Laboratory[1],
School of Information and Communication Engineering,
Sungkyunkwan University,
Chunchun-dong 300, Jangan-gu, Suwon, Kyunggi-do,
Republic of Korea
{tkkim,dylee}@rtlab.skku.ac.kr, tmchung@ece.skku.ac.kr
[2]Korea Institute of Science and Technology Information
ohbyeon@kisti.re.kr

Abstract. With the growing number of attacks on network infrastructures, we need better techniques to detect and prevent these attacks. Each security system in the distributed network requires different security rules to protect from these attacks efficiently. So the propagation of security rules is needed. Therefore, we introduce mobile agents that propagate security rules by constantly moving around the Internet as a solution to propagation of security rules. This paper describes a new approach for propagation of security rules in large-scale networks, in which mobile agent mechanisms are used. To evaluate the proposed approach, we simulated a policy propagation model using a NS-2 (Network Simulator). Our new approach presents advantages in terms of spreading rules rapidly and increasing scalability.

1 Introduction

Significant progress has been made in the improvement of computer system security. However, attacks and invasions of this kind involving computers have become frequent. Thus, security has become a key word for most companies worldwide. Intrusion detection is defined [22] as, "The problem of identifying individuals who are using a computer system without authorization and those who have legitimate access to the system but are abusing their privileges." Intrusion-detection systems aim at detecting attacks against computer systems and networks. Approaches to detecting intrusions can be broadly classified into two categories: Anomaly Detection and Misuse Detection. Misuse detection is best suited for reliably detecting known use patterns. Misuse detection systems can detect many or all known attack patterns, but they are of little use for as yet unknown attack methods. Therefore, it is required to keep security rules up to date with new vulnerabilities and environments and distributed systems are increasingly requiring differentiated policies that govern individual and collective behavior of entities in the system. A centralized approach for managing heterogeneous intrusion detection systems has some problems. For example, managing heterogeneous systems individually requires too much work and

M.E. Orlowska et al. (Eds.): ICSOC 2003, LNCS 2910, pp. 514–526, 2003.

high cost in large-scale distributed networks. Therefore, we suggested a policy propagation model for a large-scale distributed network using mobile agents. The large-scale distributed network is divided into several security zones, and one mobile agent is assigned to one security zone to propagate security rules. Each IDS in a security zone doesn't have to contain all kinds of security rules but it must have locally specialized security rules to protect systems. The priority of each security rule is different in each IDS according to its operation. Therefore, the procedure of checking the conflict of security rules is necessary when the IDS receive the security rules from the mobile agent.

Before designing the mobile agent-based rule propagation method, we investigated patterns of the intrusion detection rules. The intrusion detection rules can be divided into two systems: A forward-chaining rule-based system and a backward-chaining rule-based system [6]. A forward-chaining rule-based system is data-driven: each fact asserted may satisfy the conditions under which new facts or conclusions are derived. Alternatively, backward-chaining rule-based systems employ the reverse strategy; starting from a proposed hypothesis they proceed to collect supportive evidence. Our proposed system uses the forward-chaining rule-based system.

This paper mainly describes a dynamic agent-based rule propagation model for improving efficiency of negotiation between mobile agents and intrusion detection systems. In section 2, some related works and a mobile agent description are shown. In section 3, the design of a mobile agent-based security rule propagation and negotiation model is clarified. In section 4, the policy propagation model is simulated with respect to time. Section 5, this paper is summarized.

2 Related Works

In this section, we overview the mobile agent and briefly describe the characteristics of typical agent-based intrusion detection systems (IDSs) – EMERALD, AAFID, IA-NSM, and IDA.

2.1 Overview of Mobile Agent

A mobile agent is a kind of independent program which can migrate from one node to another node in a distributed network by itself. Unlike the traditional distributed techniques such as the Client-Server model and Code on Demand model, a mobile agent has advantages as follows:

- proper use of existing resources to fulfill user's assignment
- debase network traffic
- balance network load
- support fault-tolerance
- support mobile user
- support customized services

A mobile agent has its own lifecycle - creating, halting, executing, service searching, arriving at a new host, migrating, returning to the original host and

terminating. We use this agent technology in misuse intrusion detection systems to spread the rule containing the intrusion information. A mobile agent consists of three parts: resource, function and rule information. Figure 1 shows the component of a mobile agent [2].

The resource section contains the hardware resource, computing environment, and encrypted hash value. The computing environment means the serialized state of the agent is encoded in the method part. The encrypted hash value is used to check the integrity of the agent. The function part includes the control module, communication module, and method. The control module controls all functions of the mobile agent including authentication and authorization. The communication module provides a secure communication channel between the agent and IDS.

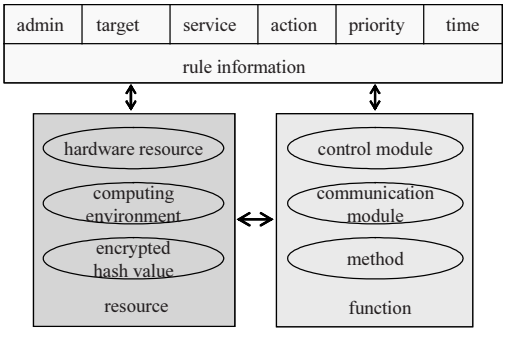


Fig. 1. Constitution of mobile Agent

A method module includes a concrete computing code and program [2]. Rule information in the agent is composed of six elements: administrator, target, service, action, priority, and time information. 'Admin' is the operator who makes the intrusion detection rule; 'target' shows the object that needs to be protected from attacks; 'service' shows which service is controlled by the intrusion detection rule; 'action' describes the action that is performed to the target; 'priority' means the order of rules; 'time' shows the generation time of security rules. Mobile agents offer a new paradigm for distributed system environments, but there are two kinds of security issues specific to a mobile agent [2, 16]: Misuse of hosts by mobile agents, misuse of mobile agents by hosts and other mobile agents. A mobile agent can abuse the information, software, hardware, or resources of a host computer. Also, a mobile agent can be destroyed, stolen from, subverted, and trapped by other mobile agents and host computers.

We have suggested a security model that provides safety from a malicious agent and a malicious host. Encrypted hash values guarantee the integrity of the mobile agent and protect unauthorized modification of the mobile agent. A trusted third party authenticates a mobile agent and host mutually. When conflict occurs between the rules of a mobile agent and those of IDS, negotiation of security rules occurs between mobile agent and IDS to solve the conflict of security rules. The use of rule adapter prevents the mobile agent and IDS from the misuse of mobile agent and IDS. Thus this system is protected from the activities of malicious agents and malicious hosts.

2.2 The Characteristics of the Typical Agent-Based IDSs

Agent technology has been applied in a variety of fields, particularly in artificial intelligence, distributed systems, software engineering and electronic commerce. Generally, an agent can be defined as a software program that can execute a complex task on behalf of the user [7]. Some authors have proposed the use of autonomous agents to construct non-monolithic intrusion detection systems [8, 9, 10]. The capacity of some autonomous agents to maintain specific information of their application domains, in this case security, confers great flexibility on these agents and hence, on the entire system. The characteristics of typical agent-based IDS are follows:

- **EMERALD**

The SRI(Stanford Research Institute) EMERALD(Event Monitoring Enabling Response to Autonomous Live Disturbance) project addresses the problems of network intrusion via TCP/IP data streams. Network surveillance monitors observe local area network traffic and submit analysis reports to an enterprise monitor, which correlates the reports. EMERALD seems to concentrate the intelligence in a central system and does not incorporate any agent technology [11, 12, 13].

- **AAFID**

The Autonomous Agents for Intrusion Detection (AAFID) project at Purdue University is based on independent entities called autonomous agents that perform distributed data collection and analysis. AAFID employs a hierarchy of agents. At the root of the hierarchy are monitors, which provide global command and control and analyze information flowing from lower level nodes. At the leaves, agents collect event information. The agents reside on special purpose agent platforms, called transceivers. Transceivers perform command and control of locally running agents and analyze and reduce the information received from the agents. Transceivers feed processed information onto monitors. Agents seem to be static once they are deployed to a transceiver, but are replaceable through reconfiguration [14, 19].

- **IA-NSM**

The Intelligent Agents for Network Security Management (IA-NSM) Project for Intrusion Detection using intelligent agent technology provides flexible integration of a multi-agent system in a classically- networked environment to enhance its protection level against inherent attacks [15].

- **IDA**

The Intrusion Detection Agent (IDA) system relies on mobile agents to trace intruders among the various hosts involved in an intrusion. IDA works by focusing on specific events that may relate to intrusions, referred to as “Marks Left by Suspected Intruder (MLSI).” If an MLSI is found, IDA gathers information related to the MLSI, analyzes the information, and decides whether or not an intrusion has occurred. The system follows a hierarchical structure, with a central manager at the root and a variety of agents at the leaves [19].

As previously mentioned, many research labs are currently working on applying agents to intrusion detection. Other works in this field are encouraging, as well. However, among all these efforts, there was no mobile agent-based rule propagation and negotiation model for intrusion detection system. In this paper, we present the model of the Mobile Agent-based Rules propagation and negotiation System (MARS) for the IDS in the large-scale distributed network environments. In MARS, the mobile agents are used as carriers, especially negotiating with other intrusion detection systems about security rules. And also, we simulate the efficiency of rule propagation model using NS-2 simulator.

3 The Design and Propagation Rules of MARS

3.1 The Design of MARS

This section describes the design details of the mobile agent-based rule propagation and negotiation system and clarifies the objectives in designing a MARS. We set up the following goals:

- Propagation rules in dynamic and distributed environments
- Security of the mobile agent-based rule propagation system
- Authentication of the mobile agent
- Negotiation of security rules between mobile agent and intrusion detection system

As shown in Figure 2, if integrated security managers managing the intrusion detection system find a new attack, they enter information about the new attacks into the security management client. Then the information is transmitted to the security management server. The security management server generates a security policy about new attacks and sends the security rule to the total policy database. The Agent Generator generates a mobile agent, which contains the security policy about new attacks. Then the mobile agent migrates to the assigned security zone through the Internet, and moves dynamically in the security zone to propagate the intrusion detection rule.

When managing heterogeneous IDSs in security domains, it is difficult to guarantee integrity for IDS policies. Breaking the integrity of IDS policies result in serious problems for network security. It also requires more efforts, and costs and managing distributed policies of network IDSs for security managers. Most IDS products currently support remote management to solve above problems. With remote management functionality, a security manager is able to manage multiple IDSs at one location. But, in many cases, remote management tools are limited to the same IDS products (or IDS products developed by one vendor). Therefore, to manage heterogeneous IDSs in a large-scale distributed network at a low cost, we have designed the MARS.

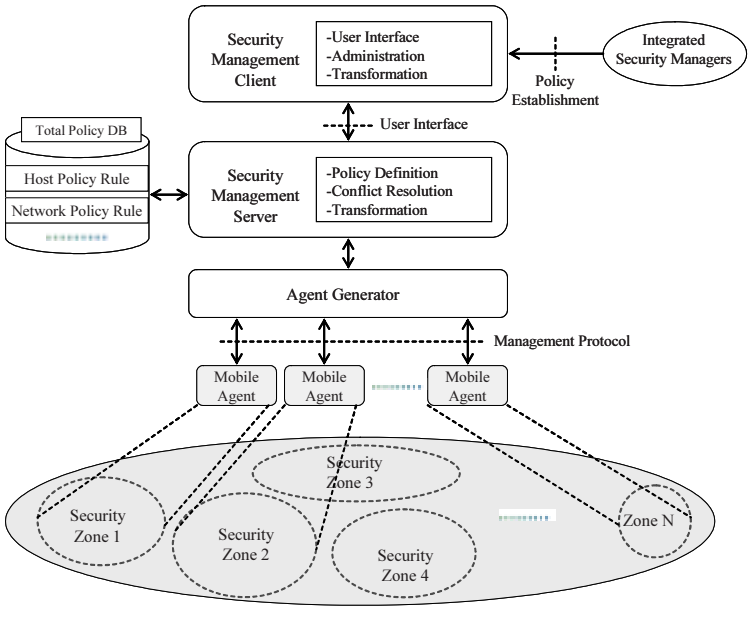


Fig. 2. Conceptual architecture of MARS

In this paper, we focus on the mobile agent having security rules and the ability of negotiation to propagate security rules to the heterogeneous IDSs in large-scale network environments. The conceptual architecture of mobile-agent based rule propagation is shown in Figure 3. The Rule adapter communicates with mobile agents and makes decisions as to whether the rule is necessary to the intrusion detection system. A trusted third party is an authentication server that authenticates the mobile agent.

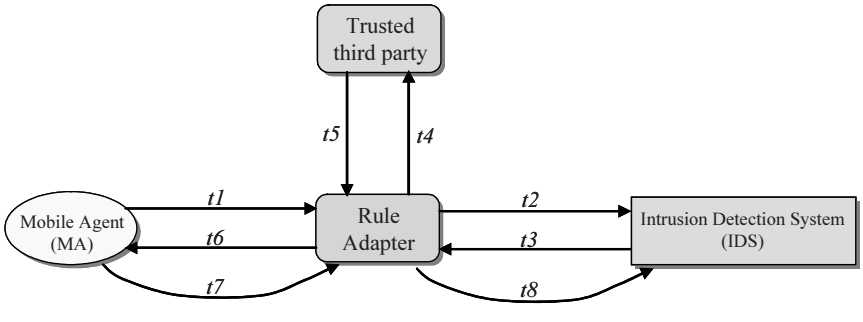


Fig. 3. Conceptual Architecture of rule propagation

When a mobile agent arrives at the rule adapter(*t1*), The Rule adapter checks the policy conflict(*t2*). If there is no conflict between the rules of the mobile agent and the rules of the intrusion detection system(*t3*), then the rule adapter authenticates the

mobile agent using the trusted thirty party(*t4-t6*). And the rule adapter checks the integrity of the rule using the hash value in the mobile agent(*t7*). Finally, the rule having the information about the new attack is added to the rules of the intrusion detection system(*t8*). When there is a conflict between the rules of the agent and of intrusion detection systems, the negotiation procedure is processed to solve the conflict problems. Therefore, the rule adapter serves as a mediator in negotiations between the mobile agent and the IDS. Also we use XML as message exchange schemes among mobile agent and IDS. As stated in [20], this scheme supports increasing scalability and simplicity to propagate security policy to the different IDSs in security zones of large scale network environments.

3.2 Negotiation Procedure of Security Rules

Whenever several policies are applied to the same target, there is potential conflict between them. A preliminary work on conflict classification is reported in [3, 4, 5] where different types of conflict are distinguished according to the overlaps among the subject, action and target scopes. In order to guarantee the integrity of policies, the rule adapter processes the integrity procedure, checking at each time when mobile agents propagate the security rules. The rule adapter performs the integrity check procedure under the following three conditions. If any of these conditions are satisfied, we can consider there is a policy conflict.

The primary function of the rule adapter is detecting and resolving policy conflicts. The policy of IDS, P(x), is defined by the existing policy(old) and the newly propagated policy(new). A policy P(x) consists of policy Target T(x), Service S(x) and the action of policy A(x). That is, P(new) and P(old) are defined as follows [1]:

$$P(new) = \{T(new), S(new), A(new)\}, P(old) = \{T(old), S(old), A(old)\}$$

Condition 1. Equivalence; P(old) = P(new)

An equivalent policy conflict occurs when two policies have the same values of policy target T(x), service S(x) and the action of policy A(x). This can be resolved by the user levels – security administrator, general user. For example, security administrator has 1-level rights, system manager has 2-level rights, and general user has 9-level rights. The policy with the highest admin level is applied when two or more policies conflict.

Condition 2. Contradiction; P(old) ↔ P(new)

A contradictable policy conflict occurs when both positive and negative policies of the same kind (i.e. permit policies or deny policies) exist. This conflict is very serious because there are no means of deciding whether the action is to be permitted or denied. A contradictable policy conflict can be resolved by user level, priority of policy, and the time of rule creation in rule information. The classification of contradiction conflict is shown in Table 1.

Table 1. Classification of contradiction conflicts

T(x); Target	S(x); Service	A(x); Action
T(old) = T(new)	S(old) = S(new)	A(new) = Positive_Action A(old) = Negative_Action
		A(new) = Negative_Action A(old) = Positive_Action
T(old) ≠ T(new)		A(new) = Positive_Action A(old) = Negative_Action
		A(new) = Negative_Action A(old) = Positive_Action

Condition 3. Inclusion; {P(old)⊃P(new) ∨ P(old)⊂P(new)} ∧ {p(old) ↔ P(new)}

Examples of inclusive policy conflict are shown in table 2. This policy conflict occurs when the inclusive relationship with contradictable relationship between existing policy P(old) and newly requested policy P(new) exist. This can be resolved by user level, priority of policy, and the time in rule information.

Table 2. Classification of inclusive policy conflicts

T(x); Target	S(x); Service	A(x); Action
T(old) = T(new)	S(new) ⊂ S(old)	A(new) = Positive_Action A(old) = Negative_Action
		A(new) = Negative_Action A(old) = Positive_Action
	S(new) ⊃ S(old)	A(new) = Positive_Action A(old) = Negative_Action
		A(new) = Negative_Action A(old) = Positive_Action
T(old) ≠ T(new)	S(new) ⊂ S(old)	A(new) = Positive_Action A(old) = Negative_Action
		A(new) = Negative_Action A(old) = Positive_Action
	S(new) ⊃ S(old)	A(new) = Positive_Action A(old) = Netative_Action
		A(new) = Negative_Action A(old) = Positive_Action

To solve the above problems, rule adapter compares the value of admin level, priority, and time in rule information between P(old) and P(new). At first, rule adapter compares the level of admin who generate the security rule. Each admin has different rights. The policy with the highest admin level is applied when two or more policies conflict. If the rule information has the same admin, then rule adapter checks the priority of each security rule. The rule having the higher priority is adapted. If two

security rules have the same priority and the same admin, the rule adapter checks the time in rule information. The newly generated security rule has higher priority than the old one. The comparison process is shown in Figure 4.

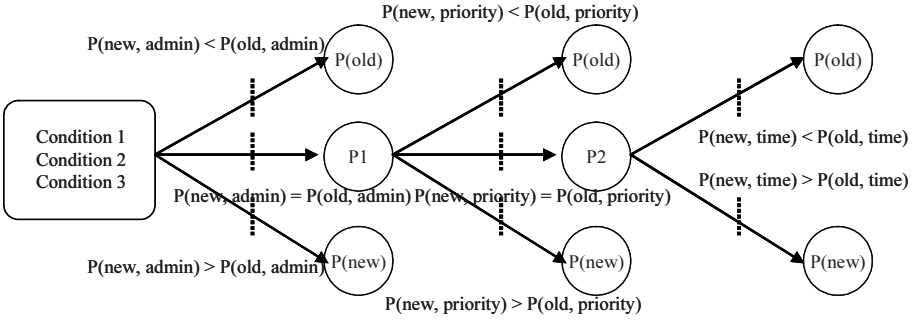


Fig. 4. The comparison process of security rules

4 Simulation Results

According to [21], delay time is the most necessary to users to handle the application. To evaluate our model with respect to time, we used the NS-2 (Network Simulator). NS-2 is an open-source simulation tool that runs on Linux. It is a discreet event simulator targeted at networking research and provides substantial support for simulation of routing, multicast protocols and IP protocols, such as UDP, TCP, RTP and SRM over wired and wireless (local and satellite) networks [17].

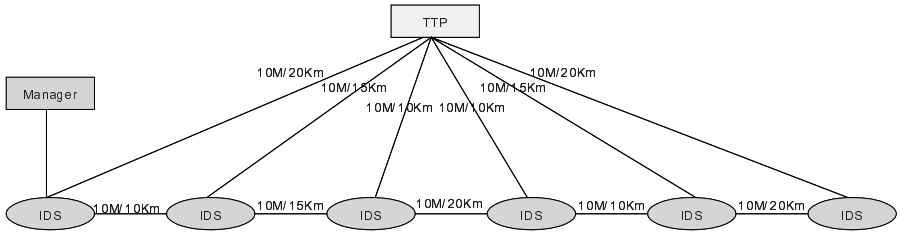
Figure 5 shows the network topology used in simulation. In (a), mobile agent moves from one IDS to another IDS to propagate the security rule whereas in (b), centralized manager transport security rules to each IDS. MARS is designed to use in a large-scale network environment and propagate the security rule to heterogeneous IDSs. Therefore, MARS use XML as message exchange schemes [20] and divide the large-scale network into several security zones to manage and assign a mobile agent. But in this simulation, we suppose that IDSs are homogeneous products developed from same vendor because the rule exchange schemes using XML(eXtensible Markup Language) are only possible in MARS. In figure 5, (a) means a security zone of MARS.

Parameters used in this simulation are like this: security rule is propagated to six IDSs; a TTP (Trusted Third Party) is used; Network bandwidth is 10Mbps; the distance from one IDS to another IDS are different respectively as shown in Figure 5; the size of rule files is 4.5kbyte.

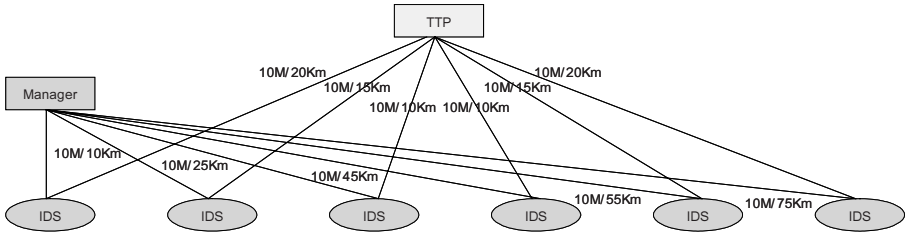
The goal of the simulations was to evaluate our MARS approach against the centralized rule propagation IDS approach with respect to time. Delay can be estimated as follows [18]:

-Delay = Propagation delay + Transmission delay + Queuing delay
-Propagation delay = Distance / 2.3×10^8 (in a cable)
-Transmission delay = Size / Bandwidth

We didn't consider the queuing delay in this simulation. Figure 6 shows the elapsed transmission time of mobile agent and rule files used in the simulation. The results show that MARS is more efficient than centralized IDS method.



(a) The simulation topology of MARS



(b) The simulation topology of centralized approach

Fig. 5. Simulation topology

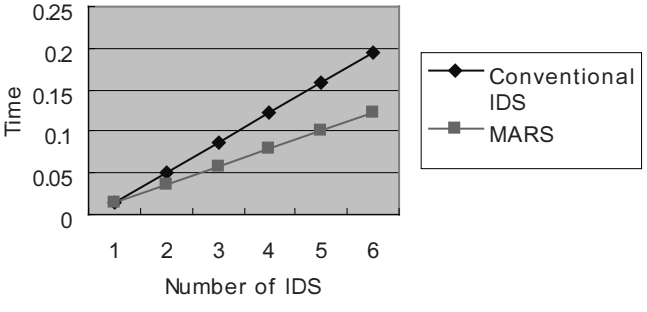


Fig. 6. Transmission elapsed time of Centralized approach and MARS

Also, we simulated this method using the tree topology. Tree topology combines characteristics of linear bus and star topologies. It consists of groups of star-configured computers connected to a linear bus backbone cable.

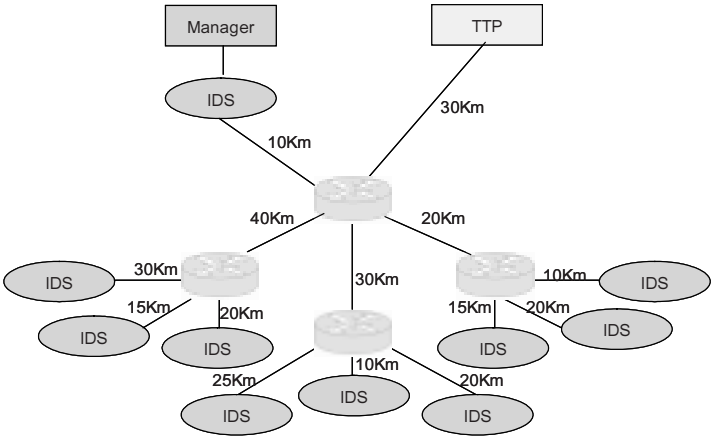


Fig. 7. Tree Simulation topology

Parameters used in this simulation were the same as above conditions. Figure 8 shows the elapsed transmission time of mobile agent and rule files. Also, the results of this simulation showed that the method of MARS was more efficient than that of centralized IDS.

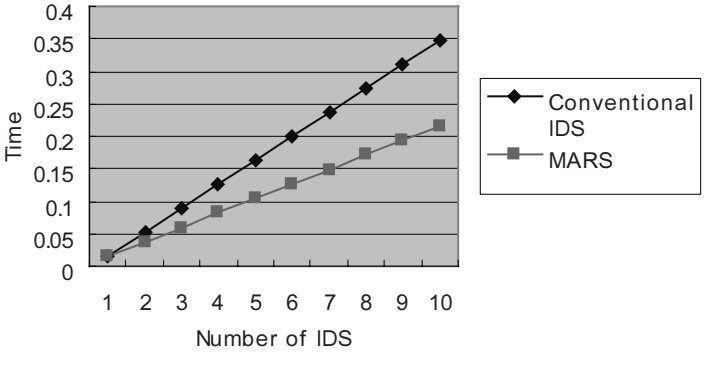


Fig. 8. Transmission elapsed time of Centralized approach and MARS in Tree topology

5 Conclusion and Future Works

With the growing number of attacks on network infrastructures, we need better techniques to detect and prevent these attacks. To protect networks from intrusions

and attacks, many security systems have been developed such as firewalls and intrusion detection systems. These days, we have large-scaled networks and various security systems. To protect systems from attacks efficiently, the propagation and negotiation of security rules are required.

The goal of this research was to suggest a more proactive mobile agent-based rule propagation and negotiation model. The mobile agent can move around the IDSs within the divided security zone and quickly propagate the security rules using the XML message exchange scheme. And the mobile agent can negotiate with IDS to select, reject, upload and download the security rules. Also, mutual authentication among mobile agent, trust third party, and IDS was suggested to establish identity and maintain its own security context. And the results of the simulation show that MARS take less time than the conventional IDS to propagate the security rules. Therefore, MARS can present advantages in terms of spreading rules rapidly and increasing scalability.

In future work, we plan to improve the functions of the mobile agent so as to cooperate with other security systems about security policies and intrusion detection.

References

1. Dong-Young Lee, Dong-Soo Kim, Tae-Kyung Kim, Tai M. Chung, "Centralized Approach for Managing Heterogeneous Firewalls in Distributed Network Environments," WISA2002, Aug. 2002.
2. L. Qi, L. Yu. "Mobile agent-based security model for distributed system," Systems, Man, and Cybernetics, 2001 IEEE International Conference, 2001.
3. J. Moffett, Morris S. Sloman, "Policy Conflict Analysis in Distributed System Management," Journal of Organizational Computing, Vol.4, No.1, pp.1–22, 1994.
4. Emil C. Lupu, Morris Sloman, "Conflicts in Policy-Based Distributed Systems Management," Journal of IEEE Transaction on Software Engineering, Vol. 25. No.6, pp.852–869, 1999.
5. E. Lupu, M. Sloman, "Conflict Analysis for Management Policies," International Symposium on Integrated Network Management IM'97, pp.430–443, 1997.
6. U. Lindqvist and P. A. Porras. Detecting computer and network misuse through the Production-Based Expert System Toolset (PBEST) In Proceedings of the 1999 Symposium on Security and Privacy, Oakland, California, May 1999.
7. H. S. Nwana. Software Agents: an Overview. Knowledge Engineering Review, 1996.
8. M. Crosbie and G. H. Spafford. Defending a Computer System using Autonomous Agents. Technical Report No. 95–022, Dept. of Comp. Sciences, Purdue University, March 1996.
9. M. Crosbie, and E. H. Spafford. "Active Defense of a Computer System using Autonomous Agents," Technical Report CSD-TR-95-008, Department of Computer Sciences, Purdue University, 1995.
10. Balasubramaniyan, Jai, J. O. Garcia-Fernandez, E. H. Spafford, and D. Zamboni. An Architecture for Intrusion Detection using Autonomous Agents. Department of Computer Sciences, Purdue University; Coast TR 98-05; 1998.
11. G. G. Helmer, J. S. K. Wong, V. Honavar, and L. Miller. Intelligent agents for intrusion detection. In Proceedings, IEEE Information Technology Conference, pages 121–124, Syracuse, NY, September 1998.

12. A. Porras and P. G. Neumann. EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances. In Proceedings of the National Information Systems Security Conference, Oct 1997.
13. A. Porras and A. Valdes. "Live Traffic Analysis of TCP/IP Gateways," Networks and Distributed Systems Security Symposium, March 1998.
14. B. Jai, J. O. Garcia-Fernandez, E. H. Spafford, and D. Zamboni. An Architecture for Intrusion Detection using Autonomous Agents. Department of Computer Sciences, Purdue University; Coast TR 98-05; 1998.
15. K. Boudaoud, H. Labiod, R. Boutaba, Z. Guessoum. Network security management with intelligent agents. Network Operations and Management Symposium, 2000. NOMS 2000.
16. S. Greenberg, C. Byington, T. Holding, G. Harper, "Mobile Agents and Security," IEEE Communications Magazine, July 1998.
17. NS network simulator. http://www-mash.cs.berkeley.edu/ns.
18. L. Peterson and B. Davie. Computer Networks: A Systems Approach. Morgan Kaufman, 2000. 2nd Edition.
19. W. Jansen, P. Mell, T. Karygiannis, D. Marks, Applying Mobile Agents to Intrusion Detection and Response, October 1999.
20. Kwang H. Kim, Tae-Kyung Kim, Dong S. Kim, Tai M. Chung, "The Design of XML-based Internet Security Integrated System Architecture", International Conference on Computational Science 2003 (ICCS 2003), June 2003.
21. NSF CISE Grand Challenge in e-Science Workshop Report, http://www.evl.uic.edu /activity/NSF/index.html, Jan 24, 2002.
22. B. Mukherjee, T. L. Heberlein and K. N. Levitt. "Network Intrusion Detection," IEEE Network, May/June 1994.