AGENT ORIENTED MODELING FOR PROXY SERVER

FAZIDAH BT OTHMAN

Tesis ini dikemukakan sebagai
memenuhi syarat penganugerahan
Ijazah Sarjana Sains (Sains Komputer)

Fakulti Sains Komputer dan Sistem Maklumat
Universiti Teknologi Malaysia

APRIL. 2004

DEDICATION

This thesis is dedicated to my beloved husband who stands all the time besides me, supports me and encourages me to finish up my work;

For my parents who always there for me…
Ayahanda Othman Mustafa and Bonda Nyapon Abdul Wahab;
Thank you for everything and I love you.

All time sisters and brothers..
Kak ina, abang an, abang edi & ana;
You are the best.

My warmest thoughts for all my friends..
Kak shidah, kak aya, linda & nik;
You are always in my mind.

COMPLIMENTARY

My special thanks to:

PROFESSOR DR. ABDUL HANAN ABDULLAH
Supervisor

FAMILY & FRIENDS

FSKSM

SPS

Thank you.

ABSTRAK

Pelayan proksi pintar merupakan teknologi baru untuk sistem Dinding Api. Pelayan proksi jenis ini dapat melakukan lebih banyak daripada hanya memaparkan semula permintaan. Ia menggunakan ejen untuk konsep capaian bagi mendapatkan penghantaran yang lebih baik antara pelayan dan pelanggan seperti proksi CERN HTTP dan pelayan proksi Microsoft. Ejen digunakan untuk tujuan carian dan mendapatkan maklumat bagi menangani semua permintaan dokumen dari luar. Pelayan proksi itu sendiri bukan merupakan ejen sebaliknya pembangun membangunkan ejen tersebut secara berasingan dan disertakan sebagai salah satu fungsi di dalam sistem Dinding Api sedia ada. Berbeza dengan 'Proxy Agent' di dalam penyelidikan ini, ia membangunkan pelayan proksi itu sebagai ejen dan bertujuan untuk memperbaiki masalah konfigurasi. Senibina SIGAL telah digunakan yang mana ia berasaskan kepada senibina pelayan/pelanggan. Senibina ini telah diperbaharui untuk memenuhi keperluan pelayan proksi yang menggunakan TIS FWTK. Jenis ejen yang digunakan ditukar untuk menjadi ejen pintar dan untuk implementasinya, teknik CBR telah digabungkan dalam proses pembangunan ejen ini. Ejen ini telah diprogramkan untuk memiliki ciri-ciri adaptasi dan pembelajaran. Untuk membangunkan satu sistem pelbagai ejen seperti yang dimaksudkan itu, satu metodologi yang berstruktur perlu diikuti dan MaSE telah dipilih untuk permodelan 'Proxy Agent'. Penyelidikan telah berjaya menghasilkan permodelan yang lengkap bagi 'Proxy Agent' dan telah diimplemen serta diuji menggunakan AgentTool yang dibangunkan bersama MaSE. Senibina ini diuji menggunakan persekitaran sebenar pelayan proksi dan keputusan menunjukkan bahawa ia melakukan tugas konfigurasi (konfigurasi rangkaian) dengan lebih baik daripada konfigurasi secara manual yang seringkali berlaku kecuaian dan kesilapan.

ABSTRACT

Intelligent proxy server is one of the new technologies for firewall system. This type of proxy can do a great deal more than simply relay requests. They use agent to implement caching concept in order to get better transmission between client and server like CERN HTTP proxy and Microsoft Proxy Server. The agent was meant for search and information retrieval which implements caching concept and handle all request for remote documents. The proxy server itself is not an agent; developers developed the agent separately and incorporate the agent into the existing firewall. Compared to Proxy Agent in this research, the proxy server was developed as an agent and was meant for fixing the configuration problems. It follows from SIGAL's architecture, which based on client/server. The improvement towards the architecture has been made to fulfill the TIS FWTK proxy server. The type of agent used in this architecture has been changed into intelligent agent and to implement this, the CBR technique has been incorporate into the development process. Thus the agent has been programmed to have adaptation and learning features. To develop such multiagent system, a structured method must be followed and therefore MaSE has been chosen to model the Proxy Agent framework. This research has provided with a step-by-step modeling process for Proxy Agent and has been implemented and tested using AgentTool, which comes together with MaSE. The Proxy Agent framework has also been tested to be used in real proxy server environment and the result shows that the Proxy Agent has better performance in performing the configuration (network configuration) compared to manual configuration which always cause misconfiguration.

TABLE OF CONTENTS

LIST OF TABLE

LIST OF FIGURE

LIST OF ABBREVIATION

AgDL	-	Agent Definition Language
AgML	-	Agent Modeling Language
BDI	-	Belief, Desire and Intention
CBR	-	Case Base Reasoning
FTP	-	File Transfer Protocol
GDL	-	Georeferenced Digital Libraries
HTTP	-	Hyper Text Transfer Protocol
ICMP	-	Internet Control Message Protocol
ICSA	-	International Computer Security Association
IR	-	Information Retrieval
KGR	-	Kinny, Georgeff, Rao
MaSE	-	Multiagent System Engineering
MIME	-	Multipurpose Internet Mail Extensions
NN	-	Neural network
OAA	-	Open Agent Architecture
PA	-	Proxy Agent
POP	-	Post Office Protocol
SIGAL	-	French acronym for Geographic Information System and Software Agent
SMTP	-	Simple Mail Transfer Protocol
TCP/IP	-	Transmission Control Protocol / Internet Protocol
TIS FWTK	-	Trusted Information System Firewall Toolkit
UML	-	Unified Modeling Language

LIST OF APPENDIX

CHAPTER I

INTRODUCTION

1.1 Overview

Proxy is a server that sits between a client application, such as Web browser, and a real server. Practically, it is a component of firewall family. Firewall is a system that enforces an access control policy between two networks, which was created to prevent dangers of the Internet from spreading to internal network. Having a proxy will make an Internet user feel that they were talking to the real system. However, in reality it is just an illusion connection between Internet clients to the real server in private network (Grennan, 2000).

Today with the exponential growth of the Internet, computer security is a major concern. At the end of year 2000, projections found the number of Internet users in the worldwide to be 374.9 million (Natale, 2001). With this huge numbers of users, we cannot expect what would they do to the Internet. Started from this point, users tried on almost every security software or application to protect their computer. At the

meantime, vendors compete each other to find and provide the best solution towards Internet security. One of the most popular security products is proxy server.

There are many ways in implementing a proxy server. It depends on a company's condition. If they could afford to buy one, then they can have it, and if they don't, the developers need to build on their own. Unfortunately, for new developers, they will find it rather difficult which sometimes makes them feel lost and exasperated during the configuration. There are tutorials or freeware which could help them through out the configuration but it is not complete; even the freeware is not fully function. Hence, some analysis on proxy toolkits should be done in order to identify their features and ability. Developers need to know the flow of proxy application. Furthermore, developers should know not only how to setup proxy server but also how to detect on security holes. They have to retest the network to make sure the configuration is correct so as the firewall configuration. Lots of testing needs to be done. For example, proxy freeware need us to modified client software in which the administrators have to download all libraries for the program that compatible with the internal system, install the library and configure it according to the server requirements. Besides that, some proxy freeware requires us to modify a user procedure. This situation does not involve with client library. But it needs us to change the procedure of using some of the Internet services through proxy (Rabiah Ahmad, 1999). Thus, providing easy and timely access and configuration is crucial from the developers' point of view.

1.2 Background of the problem

Proxy firewalls have been in existence for a relatively short time, yet during this period the demands made for them have been dramatically changed. To adapt to those

changes, firewalls had to migrate among different platforms and operating systems. In addition, the core technology and architecture upon which they were based changed as well (Nacht, 1997).

Intelligent proxy server is one of the new technologies for proxy. This type of proxy can do a great deal more than simply relay requests. It can provide better logging and access controls than those achieved through other methods. It is based on application level proxy. They use agent to implement caching concept in order to get better transmission between client and server. Example of intelligent proxy is CERN HTTP proxy (Chankhunthod and Schwartz, 1995) which still being used until now. Most of the developers had already incorporated agent in their firewall such as Microsoft Proxy Server, Firewall-1, Netscape Proxy Server, Sun One Web Proxy Server and many more. Unfortunately, none of them used intelligent agent to solve configuration problems either during the installation such as network configuration or after the installation to find out any misconfiguration, which may lead into security breaches. The agent was meant for search and information retrieval which implements caching concept and handle all request for remote documents (Telecom Italia Learning Center, 2003). The proxy server itself is not an agent; developers developed the agent separately and incorporate the agent into the existing firewall. It is important to fixed the configuration problems because proxy configuration is an essential part of a secure computing environment. Proxy configuration acts as a security barrier. It ensures that the proxy server monitors all incoming and outgoing information between the Internet and the Intranet. This is often an integral part of security enforcement in corporate firewalls within Intranet setups.

Research had indicated that various public proclamations about penetration tests show that well over half of the firewall regularly sampled is not properly configured including proxy products (Newman et al, 2000; Caminada et al, 1998; Yasin, 1998). A firewall test is first and foremost a type of penetration test. A penetration test uses

techniques designed to defeat and bypass security mechanisms to determine the effectiveness of such mechanisms (Schultz, 1996). They need modification and extra configuration during the setup especially application firewall like proxy server. Application firewalls, on the other hand, are add-on systems that are typically more difficult to manage individually, making them more expensive (Power, 1996).

Let us consider a very simple network environment in which there is a user or new developers trying to configure a simple network firewall with no expertise in Linux operating system and TCP/IP. At the meantime, they need to retest the network configuration over and over again to make sure the setup is correct. For example, in preparing the firewall environment, while installing Linux operating system, they need to do lots of checking through the files and modification and commands to be run to check the installation for network configuration. They have to refer to few files to check the status, modify it, and run it again. Besides that, they also need to do lots of other testing while configuration such as compiling kernel (Linux), configure network cards, configure network address, test network; ping inside and outside address, look at default gateway setting, telnet and many more. Basically, this entire configuration needs to be done manually. Any misconfiguration may lead into security breaches whilst firewall as a security software application should not have this careless mistake. In order to prevent this from happened, we need something that can do the configuration automatically especially the network testing part for example by using intelligent agent. Anyway, in order to implement it, research must be done to identify what type of proxy server suitable, what kind of agent architecture will suit the proxy server and what type of techniques to be used to make the agent intelligent.

Table below (Table 1.1) shows the examples of misconfiguration usually done by developers during the proxy configuration.

Table 1.1: Proxy Misconfiguration

Correct configuration	Misconfiguration
➢ Copy file Makefile.Config.Linux to Makefile.Config. Changed sources directory, destination directory, library for some services, database name and destination for binary file. ➢ Run command 'Make install'	➢ Copy to the same file name. ➢ No changes being made. ➢ Run command 'Fixmake' which damaged the whole file system.
➢ Make changes in directory /etc/inetd.config. Add all the proxy functions. ➢ Turn off echo, discard, daytime, chargen, ftp, gopher, shell, login, exec, talk, ntalk, pop-2, pop-3, netstat, systat, tftp, bootp, finger, cfinger, time, swat and linuxconfig.	➢ No changes being made or ➢ Proxy function left over or ➢ Forget to turn off the functions leaving the system to be open.
➢ Make changes in directory /etc/services. Add service name and port number.	➢ No changes being made or ➢ Wrong service name or ➢ Wrong port number
➢ Make changes in directory /usr/local/etc/netperm-table. Permit or deny the IP and host name.	➢ No changes being made or ➢ Permit all or some dangerous IP and host name which can cause the system to be open to everyone or ➢ Deny some important IP and host name, which can cause the system for not functioning properly.
➢ Do not allow localhost logins or use TCP wrappers programs to restrict the IP address and host connected.	➢ Allow localhost logins and do not use the TCP wrappers for host restriction and limited access, which can cause illegal access from outsiders.
➢ Disallow proxy access from the external interface.	➢ Forget to block outside access, which can cause hackers to make use the server to launch attack.

Figure 1.1 depicts an abstract view of the problem domain.

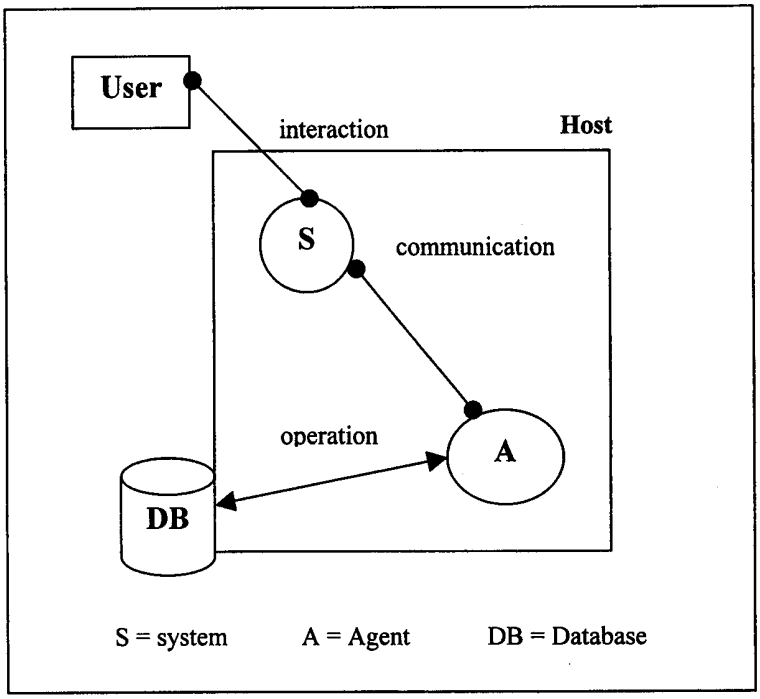

Figure 1.1: System diagram of problem domain

1.3 Statement of the problem

To develop an intelligent agent for proxy server to overcome the drawback of proxy misconfiguration as mentioned in the previous section, the problem must be identified or specified. First, started from configuration problem which mainly is network setup, what kind of technique is suitable to solve the problem, meaning that the technique must be able to identify incorrect configuration and the loopholes (after installation), then the technique must be able to make the agent reconfigure the mistakes or misconfiguration

based on the information gathered throughout the configuration process. It means that the agent must be intelligent. Secondly, in order to develop that kind of agent, there is a need to follow a certain methodology, which specifically meant for developing multiagent system. And lastly, to achieve that solution, suitable agent architecture must be identified to fulfill the proxy server environment.

As a result, the problem statement would be as follows:

i. *What kind of methodology suitable in order to design and model the agent;*
ii. *What is the suitable technique can be used to solve the configuration problem which met the intelligent features;*
iii. *What type of architecture suit both agent and proxy server environment.*

1.4 Importance of the research

This research had come out with an improved proxy agent architecture, which produces an agent solving problem approach to deal with the complexity of configuration and setup difficulties in building up a proxy. The architecture follows from software agent-oriented frameworks of SIGAL project. Enhancement had been made towards the architecture in order to fulfill the firewall environment. The technique selected, which is Case Base Reasoning (CBR), had contributed into adding intelligent features to the agent. The technique helped very much to solve the misconfiguration as it consists of adaptive and learning components. This type of configuration, called Proxy Agent, is an improvement over the earlier scheme (manual

configuration). The use of agent previously in many other fields to solve problems had proved that the use of agent could reduce human error configuration in avoiding security holes and also insufficient quality of many firewalls (Caminada et al, 1998; Yasin, 1998). As a conclusion, the contribution can be stated as follows:

i. Provide complete design and modeling process of Proxy Agent, which follows from Multiagent System Engineering (MaSE) methodology.
ii. CBR approach to solve problems.
iii. Improved software agent-oriented frameworks, which is a client/server approach for proxy server. An improved SIGAL's architecture.

1.5 Research objectives

In this research, the main objectives are:

i. To model an agent-oriented proxy server in terms of proxy setup and configuration using agent technology by including intelligent features
ii. To improve the process of proxy configuration in terms of problem solving by using CBR approach.
iii. To improve the architecture of SIGAL's to suit the problem domain of proxy agent.

1.6 Scope of the research

The scope of the research are stated below:

i. The designs were based on Multiagent System Engineering (MaSE) methodology.

ii. The architecture was based on software agent-oriented frameworks of SIGAL project (Maamar and Moulin, 2000).

iii. The firewall system is a proxy server based on TIS Firewall Toolkit (TIS FWTK) and it is a client/server approach.

iv. The configuration part is the network configuration prepared for proxy server that is based on Linux platform.

v. The development of the agent is only the framework and strictly focused on the modeling, design and architecture. Technique used for the agent will not be discussed in detail.

1.7 Summary

By creating an agent to handle the task, most of the work is automated. Agents are able to work on their own because they are not controlled directly by humans or others, cooperative which implies communication, perceptive and pro-active, which means they exhibit goal-directed behavior. Using an agent also tend to reduce network bandwidth consumption (Muller, 1997) because instead of the sending the packets to collect information throughout the network, only agent need to be sent out and finish the task on site. The Proxy Agent was designed and modeled to be able to solve the misconfiguration independently. Using the CBR approach had proved this. The agent will automatically prepare the firewall environment in a host before continuing with the

proxy server setup to solve problems from start to finish. To create such agent, the design and modeling had followed specific methodology, which is Multiagent System Engineering (MaSE) methodology. As the methodology was based on UML approach, the frameworks selected must also be compatible with UML thus resulting in the improvement of software agent-oriented frameworks as the architecture.

1.8 Thesis Organization

This thesis consists of 6 chapters and each chapter was described as stated below:

i. Chapter I explain about the background, objectives and the importance of configuration problem in proxy server setup.

ii. Chapter II discussed more about the firewall system and the existing approach being used for them and also the suggested approach being used in the development to overcome the configuration problem.

iii. Chapter III elaborate about the methodology used to overcome the configuration problem.

iv. Chapter IV explained about the proxy agent modeling, design and development.

v. Chapter V discussed about the implementation and analysis of the proxy agent and also the testing result.

vi. Chapter VI discussed about the conclusion from the implementation result, future works regarding the agent approach for proxy server and research contribution.

BIBLIOGRAPHY

Aamodt, A. and Plaza, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations and System Approaches. *Journal of AICom – Artificial Intelligence Communications, IOS Press*. 1. Vol. 7. 39-59.

Bradshaw, J.M. (1997). *Software Agents*. CA.: AAAI Press/The MIT Press.

Brazier, F., Jonker C. and Treur, J. (1998). Principles of Compositional Multiagent System Development. *Proceedings of IFIP '98 Conference*. Vienna.

Caminada, R. Reit, A.V.D. and Zanten, L.V.D. (1998). Effectiveness Firewall. *Journal of Computers and Security*. 5. Vol. 17. 425 - 426.

Chankhunthod, A. and Schwartz, M.F. (1995). *A Hierarchical Internat Object Cache*. Technical Report. Department of Computer Science, University of Colorado. CU-CS-766-95.

Chapman, B. and Zwickey, D. (1995). *Building Internet Firewall*. Suite A Sebastapol, CA.: O'Reilly Association Inc.

Chen, H. (1993). Machine Learning for Information Retrieval: Neural Networks, Symbolic Learning and Genetic Algorithms. *Journal of the American Society for Information Science*.

Cheswick, B. (1990). The Design of a Secure Internet Gateway. *Proc. of USENIX Conference*. Anaheim, CA: USENIX.

Cheswick William R. and Bellovin Steven M. (1994) *Firewalls and Internet Security – Repelling the Wiley Hacker*. : Addison Wesley, MA.

Cheyer, A. and Julia, L. (1995). Multimodal Maps: An Agent-Based Approach. *Proceedings of the International Conference on Cooperative Multimodal Communication (CMC/95)*. Eindhoven, Netherlands.

Cheyer, A. and Julia, L. (1998). MVIEWS: Multimodal Tools for the Video Analyst. *Proceedings of the 1998 International Conference on Intelligent User Interfaces (IUI98)*. San Francisco, California.

Cheyer, A. and Martin, A. (2001). The Open Agent Architecture. *Journal of Autonomous Agents and Multi-Agent Systems*. 1. Vol. 4. 143-148.

Chiariglione, L. Foundation for Intelligent Physical Agents. [online] available at www.cselt.it/fipa/fipa_rationale.htm 11 August 2000.

Clarke, E. and Wing, J. (1996). Formal Methods: State of The Art and Future Directions. *Journal of ACM Computing Surveys*. 4. Vol. 28.

Cohen, P.R., Cheyer, A.J., Wang, M. and Baeg, S. C. (1994). An Open Agent Architecture. *Proceedings of the AAAI Spring Symposium Series on Software Agents*. Satnford, California.

Cox, M. (2000). Toward Agent-Based Mixed-Initiative Interfaces. *Proceedings of the 2000 International Conference on Artificial Intelligence*. CSREA Press.

DeLoach, S.A. (1999). Multiagent Systems Engineering: A Methodology and Language for Designing Agent Systems. *Proc. of Agent Oriented Information Systems '99 (AOIS'99)*. Seattle WA. 45 – 57.

DeLoach, S.A. and Wood, M.F. (2000a). *Multiagent Systems Engineering: The Analysis Phase*. Technical Report, Air Force Institute of Technology. AFIT/EN-TR-00-02.

DeLoach, S.A. and Wood, M.F. (2000b). Developing Multiagent Systems with AgentTool. *Proc. of the 7th International Workshop on Agent Theories, Architectures and Languages (ATAL-2000)*. Boston, MA.

DeLoach, S.A. (2001). Analysis and Design Using MaSE and agentTool. *Proc. of the 12th Mid West Artificial Intelligence and Cognitive Science Conference (MAICS 2001)*. Miami University, Oxford, Ohio.

Eisermann, M. (2002). *Performance Tests with the Microsoft Internet Security and Acceleration (ISA) Server*. Technical Research Paper. University of Applied Sciences, Germany.

Elton, P. (1997). Linux As A Proxy Server. *Linux Journal*. 44es. Vol. 1997.

Grennan, M. Firewall and Proxy Server HOWTO. [on-line] available at metalab.unc.edu/mdw/HOWTO/Firewall-HOWTO.html. 8 March 2000.

Hancock, B. (1998). Security Views : Network Associates Introduces A New Firewall Concept : Adaptive Proxies. *Journal of Computers and Security.* Vol. 17. 567 – 568.

Highland, H.J. (1997). The Internet and Computer Security. *Journal of Computers and Security.* Vol. 16. 387 - 411.

Iglesias, C. A. (1998). A Survey of Agent-Oriented Methodologies. *Proceedings of the 5th International Workshop on Intelligent Agents V: Agent Theories, Architectures and Languages (ATAL-98)*. Paris.

Jones, K. (1998). Secure Internet Access to SAP's R/3: Keeping Dragons Out. *International Journal Of Nework Management*. Vol 8. 191-199.

Khalid Al Tawil and Ibrahim A. Al Khaltam. (1999). Evaluation and Testing of Internet Firewalls. *International Journal of Network Management*. Vol. 9. 135 – 149.

Kinny, D., Georgeff, M., and Rao, A. (1996). A Methodology and Modelling Technique for Systems of BDI Agents. Agents Breaking Away. *Proceedings of 7th European Workshop on Modelling Autonomous Agents in a Multiagent World (MAAMAW'96)*. Lecture Notes in Artficial Intelligence, Vol. 1038. Springer-Verlag, Berlin Heidelberg (1996) 56-71.

Kolodner, J.L. (1993). *Case-Based Reasoning*. San Mateo, CA.: Morgan Kaufmann Publishers, Inc.

Kolodner, J.L. and Leake, D.B. (1996). *Case-Based Reasoning: Experiences, Lessons and Future Directions*. Menlo Park.: AAAI Press.

Kotz, D. and Gray, R.S. (1999). *Mobile Agents and The Future Of The Internet*. White paper for Department of Computer Science. Dartmouth College. [online] available at www.cs.dartmouth.edu/~dfk/papers/kotz:future2/ 10 March 2000.

Leake, D. B. (1995). Combining Rules and Cases to Learn Case Adaptation. *Proceedings of the 17th Annual Conference of the Cognitive Science Society.*

Maamar, Z. and Moulin, B. (1997). Interoperability of Distributed and Heterogeneous Systems Based on Software Agent-Oriented Frameworks. *Proceedings of International Workshop on Cooperative Information Agents'97 (CIA'97)*. Germany.

Maamar, Z. and Moulin, B. (2000). An Overview of Software Agent-Oriented Frameworks. *Journal of ACM*. 00360-0300/00/0300es.

Martin, D. L., Cheyer, A. and Moran, D. B. (1999). The Open Agent Architecture: A Framework for Building Distributed Software Systems. *Journal of Applied Artificial Intelligence*. 1-2. Vol. 13. 21-128.

Merwe, J.V.D. and Solms, H.S.D. (1998). Electronic Commerce with Secure Intelligent Trade Agents. *Journal of Computers and Security.* Vol. 17. 435 – 446.

Muller, N.J. (1997). Improving Network Operations With Intelligent Agents. *International Journal of Network Management*. Vol.7. 116-126.

Nacht, M. (1997). The Spectrum of Modern Firewalls. *Journal of Computers and Security.* Vol. 16. 54 - 56.

Natale, W. (2001). TCP/IP And Security Software Applications. *Journal Of Computing In Small Colleges*. 3. Vol. 16. 205-211.

Newman, D., Holzbaur, H. and Bishop, K. (2000) Lab Test : Firewall : Don't Get Burned. *Journal of Computers and Security*. Vol. 16. 123.

Nwana, H. and Ndumu, D. (1997). *An Introduction to Agent Technology*. In Lecture Notes in Artificial Intelligence 1198: Software Agents and Soft Computing: Towards Enhancing Machine Intelligence, pages 3-26. Springer.

Odell, J., Parunak, H.V. D. and Bauer, B. (2000). Extending UML for Agents. *Proceedings of the Agent-Oriented Information Systems (AOIS) Workshop in 17th National Conference on Artificial Intelligence (AAAI 2000)*. Austin, Texas.

Oppliger, R. (1997). Internet Security: Firewalls and Beyond. *Communications Of The ACM Journal*. 5. Vol. 40. 92 - 102.

Pfleeger, C.P. (1997). *Security In Computing.* Uppersaddle River, N.J.: Prentice Hall International. Inc.

Power, R. (1996). What kind of firewall should you buy. *Journal of Computer Security.* 2. Vol. 11.

Rabiah Ahmad (1999). *Proxy System : Analysis In Concept, Design and Implementation.* Royal Holloway University: Thesis. M.Sc.

Ramarapu, N. and Raisinghani, M. S. (1997). Integration of Case-Based Reasoning with Object-Oriented Database Management Systems For Efficient Management of Large Casebases. *Proceedings of 3rd Americas Conference on Information System.* Indianapolis, Indiana.

Ranum, M.J. (1993). Thinking About Firewalls. *Proc. of the 2nd World Conference on System Management and Security (SANSII).*

Sample, C. (1998). Kicking Firewall Tires. *Journal of Computers and Security.* Vol. 17. 223.

Schultz, E.E. (1996). How to Perform Effective Firewall Testing. *Journal of Computer Security.* 1.Vol. 12.

Schumacher, J., Wilke, W. and Smyth, B. (1998). Domain Independent Adaptations using Configuration Techniques. *Journal of NEC Research Institute.*

Sooyong, P., Kim, J. and Lee, S. (2000). Agent-Oriented Software Modeling with UML Approach. *Journal of IEICE Trans. Information & System.* 8. Vol. E83-D. 1631-1641.

Sparkman, C.H., DeLoach, S. A. and Self,, A. L. (2001). Automated Derivation of Complex Agent Architectures from Analysis Specifications. *Proceedings of 2nd International Workshop on Agent-Oriented Software Engineering (AOSE-2001).* Montreal, Canada. *ACM.* 1-58113-000-0.

Stone, P. (1996). Single Agent vs. Multiagent. *Journal of the IEEE Transactions on Knowledge and Data Engineering (TKDE).* Computer Science Department, Carnegie Mellon University, Pittsburgh.

Stone, P. and Veloso, M. (1996). Multiagent Systems : A Survey from Machine Learning Perspective. *Journal of the IEEE Transactions on Knowledge and Data Engineering (TKDE).* Computer Science Department, Carnegie Mellon University, Pittsburgh.

Sundaram, A. (2000). An Introduction to Intrusion Detection. *Journal Of ACM.* Vol. 3.

Suraya, dan Azlina. (1998). *Penghantaran Mel dan Web Melalui Firewall Dual Homed Host.* Universiti Teknologi Malaysia : Thesis. B.Sc.

Sycara, K., Decker, K., Pannu, A., Williamson, A. and Zeng, D. (1996). Distributed Intelligent Agents. *Journal of IEEE Expert-Intelligent Systems and Their Applications.* 6. Vol. 11. 36-45.

Telecom Italia Learning Centre (2003). Proxy Server with Intelligent Prefetcher and Intelligent Proxy Server (IPS) for Mobile Communication System. *Proceedings of Internet and Mobile Technologies Conference.* Italy.

Tveit, A. (2001). A Survey of Agent-Oriented Software Engineering. *Proceedings of 1st NTNU Computer Science Graduate Student Conference.* Dragvoll, Trondheim.

Wack J. P. and Lisa C. J. (1995). *Keeping your site comfortably secure : An Introduction to Internet Firewalls*. Special Publication 800-10, National Institute of Standards and Technology.

Wood, M. and DeLoach, S.A. (2000b). An Overview of the Multiagent Systems Engineering Methodology. *Proc. of the 1st International Workshop on Agent-Oriented Software Engineering (AOSE-2000)*. Limerick, Ireland.

Wooldridge, M., Jennings, N. R. and Kinny, D. (2000). The Gaia Methodology for Agent-Oriented Analysis and Design. *Journal of Autonomous and Multi-Agent Systems*. 3. 285-312.

Yasin, R. (1998). Burned By Your Firewall. *Journal of Computers and Security*. 3. Vol. 17. 224.

Yuhang, W. (2000). *Methodologies for Agent-Based System Analysis and Design*. Technical Research Paper. University of Calgary. SENG 609.22.