# 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.<br>➢ Run command ‘Make install’ | ➢ Copy to the same file name.<br>➢ No changes being made.<br>➢ Run command ‘Fixmake’ which damaged the whole file system. |
| ➢ Make changes in directory /etc/inetd.config. Add all the proxy functions.<br>➢ 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<br>➢ Proxy function left over or<br>➢ 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<br>➢ Wrong service name or<br>➢ 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<br>➢ Permit all or some dangerous IP and host name which can cause the system to be open to everyone or<br>➢ 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.

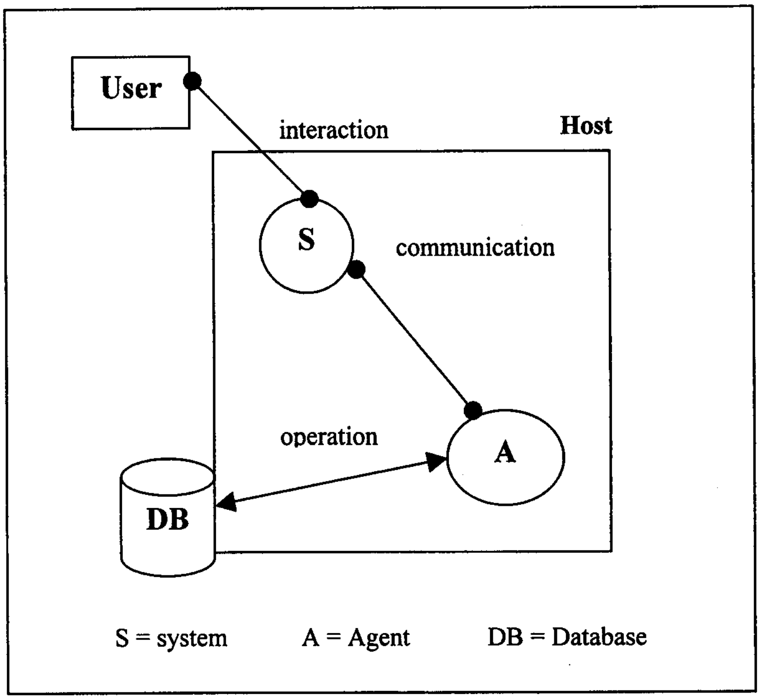

**Figure 1.1: System diagram of problem domain**

## 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.