IACSIT International Journal of Engineering and Technology Vol. 2, No.1, February, 2010
ISSN: 1793-8236

Design and Architecture for Efficient Load Balancing with Security Using Mobile Agents

R. Ezumalai, G. Aghila and R. Rajalakshmi

Abstract—**Load balancing system is commonly used for highly efficient utilization of the physical or logical resources and enhancing the performance of the distributed systems and scalability of the Internet. Numerous proposals exist for load balancing system in peer-to-peer networks, but it does not mainly address the security issues. Load balancing among the peers is critical to provide a solution for distribution of resources with security. These paper propose the three methodology for addressing an architecture for load balancing system with security. First, it address an architecture for mobile agent to roam all the nodes in an distributed network and. second, it address an architecture to rearrange the loads among the peers for better performance of the distributed system. Third, and perhaps more significantly, address the mobile agent to provide the security in a network.**

Index Terms—**Structured peer-to-peer system, Load balancing system, Mobile Agent, Intrusion detection system, Virtual Server Reassignment problem.**

I. Introduction

Distributed computing deals with hardware and software systems containing more than one processing element or storage element, concurrent processes, or multiple programs, running under a loosely or tightly controlled regime [2]. In distributed computing a program is split up into parts that run simultaneously on multiple computers communicating over a network. Distributed computing is a form of parallel computing, but parallel computing is most commonly used to describe program parts running simultaneously on multiple processors in the same computer [9]. Both types of processing require dividing a program into parts that can run simultaneously, but distributed programs often must deal with heterogeneous environments, network links of varying latencies, and unpredictable failures in the network or the computers. Load balancing is a method to distribute the work between two or more computers, network links, CPUs, hard drives, or other resources, in order to get efficient resource utilization, maximize throughput, and minimize response time. With the multiple components, instead of a single component, may increase reliability through redundancy for load balancing. The balancing service is usually provided by a dedicated program or hardware device (such as a multilayer switch). Most common applications of load balancing is to provide a Internet service from multiple servers [4]. Commonly load- balanced systems include popular web sites, large Internet Relay Chat networks, high-bandwidth File Transfer Protocol sites, NNTP servers and DNS servers.For Internet services, the load balancer is usually a software program which is listening on the port where external clients connect to access services.

PEER-TO-PEER (P2P) systems make it possible to hardness resources such as the storage, bandwidth, and computing power of large populations of networked computers in a cost-effective manner. In structured P2P systems, data items are spread across distributed computers (nodes), and the location of each item is determined in a decentralized manner using a distributed hash lookup table (DHT) [1]. Structured P2P systems based on the DHT mechanism have proven to be an effective design for resource sharing on a global scale and on top of which many applications have been designed such as file sharing, distributed file systems [2], real-time streaming, and distributed processing [6] and In computer science, a software agent is a piece of software that acts for a user or other program in a relationship of agency.Such "action on behalf of" implies the authority to decide which (and if) action is appropriate. The idea is that agents are not strictly invoked for a task, but activate themselves. A Multi Agent system is a system composed of multiple interacting intelligent agents. Multi-agent systems can be used to solve problems which are difficult or impossible for an individual agent or monolithic system to solve. Examples of problems which are appropriate to multi-agent systems research include online trading, disaster response, and modelling social structures.

The agents in a multi-agent system have several important characteristics:

1) *Autonomy:* Agents are at least partially autonomous.
2) *Local views:* There is no agent has a full global view of the system, or the system is too complex for an agent to make practical use of such knowledge
3) *Decentralization:* In Decentralization environment, agents are to share the knowledge and information and no agent will control another agent. Multi-agent systems research refers to software agents. However, the agents in a multi-agent system could equally well be robots, humans or human teams [4]. A multi-agent system may contain combined human-agent teams. Multi-agent systems can manifest self-organization and complex behaviors even when the individual strategies of all their agents are simple. The remainder of the paper is organized as follows: Section 2 contains related work;

Department of Computer Science, Pondicherry University,Pondicherry, India. (e-mail: Ezumalai1984@gmail.com, aghilaa@yahoo.com, lakshmi22006@yahoo.co.in).

Section 3 describes our proposed approach, Section 4 describes the conclusion and future work.

.

To insert images in *Word*, position the cursor at the insertion point and either use Insert | Picture | From File or copy the image to the Windows clipboard and then Edit | Paste Special | Picture (with "Float over text" unchecked).

INTERNATIONAL JOURNAL OF COMPUTER THEORY AND ENGINEERING reserves the right to do the final formatting of your paper.

II. Related Work

There are many techniques is available to provide efficient load balancing system, but it consumes a lot of resources for distributing the work. This load balancing system does not have centralized co- coordinator for allocating the resources and security. But this architecture provides the way of rearranging the loads efficiently with security. Many techniques have been discussed below for load balancing system with merits and demerits.

CFS [7] accounts for node heterogeneity by allocating to each node some number of virtual servers proportional to the node capacity. In addition, CFS proposes a simple solution to shed the load from an overloaded node by having the overloaded node remove some of its virtual servers. However,this scheme may result in thrashing as removing some virtual servers from an overloaded node may result in another node becoming overloaded. Adler et al [9] present a DHT which provably ensures that, as nodes join the system, the ratio of loads of any two nodes is O(1) with high probability. The system is organized as a tree, with additional links for routing in a hypercube topology. A joining node considers a small set of leaf nodes of the tree and joins the system by splitting an appropriately chosen leaf. However, no analysis of node departure was given and the system does not deal with varying node capacity or object distribution.

Karger and Ruhl [10] propose algorithms which dynamically balance load among peers without using multiple virtual servers by reassigning lightly loaded nodes to be neighbors of heavily loaded nodes. However, they do not fully handle the case of heterogeneous node capacities, and while they prove bounds on maximum node utilization and load movement, it is unclear whether their techniques would be efficient in practice. Douceur and Wattenhofer [11] have proposed algorithms for replica placement in a distributed filesystem which are similarin spirit with our algorithms. However, their primary goal is to place object replicas to maximize the availability in an untrusted P2P system, while we consider the load balancing problem in a cooperative system. Triantafillou et al. [12] have recently studied the problem of load balancing in the context of content and resource management in P2P systems. However, their work considers an unstructured P2P system, in which meta-data is aggregated over a two-level hierarchy.

Grosu, D [22] propose to designing protocols for resource allocation involving selfish agents and design a truthful mechanism for solving the static load balancing problem in heterogeneous distributed systems. This paper prove that using the optimal allocation algorithm the output function admits a truthful payment scheme satisfying voluntary participation. Miron Livny [2] propose an load balancing algorithms for distributed systems that consist of a number of identical processors and a CSMA communication system are presented in this paper. Some of the properties of a multi-resource system and the balancing process are demonstrated by an analytic model. Simulation is used as a mean for studying the interdependency between the parameters of the distributed system and the behaviour of the balancing algorithm.

Yu-Kwong [20] propose an approach works by using a fuzzy logic controller which informs a client object to use the most appropriate service such that load balancing among servers is achieved. Authors chosen Jini to build our experimental middleware platform, other related techniques are implemented and compared. Extensive experiments are conducted to investigate the effectiveness of our fuzzy-decision- based algorithm, which is found to be consistently better than other approaches. Yamaguchi et al [19], propose an autonomous load balancing system for distributed servers using the active object model. This system consists of distributed servers. Each server has a load balancer whichmonitors the server load, and controls the load, communicating with other load balancers. Distributed load balancers communicate in a peer-to-peer way, and a client/server model is not suitable, the active object model is very suitable. Cardellini [18] propose Distributed Web server architectures that transparently schedule client requests offer a way to meet dynamic scalability and availability requirements. The authors review the state of the art in load balancing techniques on distributed Web-server systems, and analyze the efficiencies and limitations of the various approaches. Frank Lingen [23] propose propose a system, in which mobile agents will transport, schedule, execute and return results for heavy computational jobs submitted by handheld devices. Moreover, in this way, our system provides high throughput computing environment for hand-held devices.

Jian Feng Cui [17] propose an approach to load balancing for RFID middlewares based on Mobile Agent System. Two agents, LGA and RLBA, are designed. LGA is used to gather global workload of RFID middlewares, and RLBA executes load balancing process in case of overloading.

Many authors proposed different techniques to efficiently distribute the loads among the system. But all these methods reported to have a lot of pros and cons of its own proposal. This architecture provides a technique to rearrange the loads efficiently and security with the help of agents.

Yu-Kwong [20] propose an approach works by using a fuzzy logic controller which informs a client object to use the most appropriate service such that load balancing among servers is achieved. Authors chosen Jini to build our experimental middleware platform, other related techniques are implemented and compared. Extensive experiments are conducted to investigate the effectiveness of our

fuzzy-decision- based algorithm, which is found to be consistently better than other approaches. Yamaguchi et al [19], propose an autonomous load balancing system for distributed servers using the active object model. This system consists of distributed servers. Each server has a load balancer whichmonitors the server load, and controls the load, communicating with other load balancers. Distributed load balancers communicate in a peer-to-peer way, and a client/server model is not suitable, the active object model is very suitable. Cardellini [18] propose Distributed Web server architectures that transparently schedule client requests offer a way to meet dynamic scalability and availability requirements. The authors review the state of the art in load balancing techniques on distributed Web-server systems, and analyze the efficiencies and limitations of the various approaches. Frank Lingen [23] propose propose a system, in which mobile agents will transport, schedule, execute and return results for heavy computational jobs submitted by handheld devices. Moreover, in this way, our system provides high throughput computing environment for hand-held devices.

Jian Feng Cui [17] propose an approach to load balancing for RFID middlewares based on Mobile Agent System. Two agents, LGA and RLBA, are designed. LGA is used to gather global workload of RFID middlewares, and RLBA executes load balancing process in case of overloading.

Many authors proposed different techniques to efficiently distribute the loads among the system. But all these methods reported to have a lot of pros and cons of its own proposal. This architecture provides a technique to rearrange the loads efficiently and security with the help of agents.

III. Our Approach

An agent is a small active object, which is able to carry out activities continuously and autonomously in a particular environment. Agent is autonomous, lightweight, adaptive, and mobile. Multi agent can communicate and cooperate with each other. These qualities make agent a choice for efficient rearranged of loads in a distributed network and security architecture in sensitive distributed system.

Figure 1 shows the three essential components of the architecture: Supervisor Agent, Monitoring Agent and Locate Agent are present in this system. Monitor agent is a mobile agent which gathers information from each system and sends it to the Supervisor Agent. Supervisor agent is present in virtual server analyze the information and make a decision using many different types of algorithm [1] [2] [5]. If the supervisor agent can judge that any node have more load than the current load, it will use locate agent. Locate agent use different types of existing algorithm [3] [2] to find out neighboring node that has more load than current load. It shifted a load from this present node with all information to that respective node and it act as a virtual server.

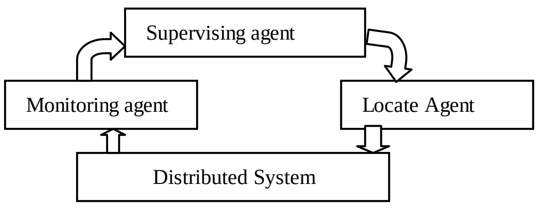

Figure1 System Architecture

a. Monitoring Agent

Monitoring agent or mobile agent can roam on each node in distributed system networks. It gathers each and every system information dynamically and sends this information to supervisor agent. In this process it gathers security side information and it monitors the environment dynamically. There are four components in this agents, which are information collections, filtration, coder and communication. Fig2 shows its components of Monitor agents.

The collection component is responsible for collecting information of all the nodes and its hand in filter component. Coder component is useful to code the individual system information by number for identification of the system. In order to avoid malicious access, to number all the nodes in the network.

Filter information is useful to filter the unwanted information of the collections. Communication is useful to communicate with the supervisor agent for further analysis of the load target system.

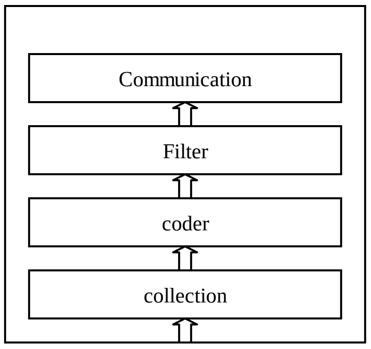

Figure 2 Monitoring Agent

b. Supervisor Agent

Core of the architecture is the supervisor agent. It collects the information from monitoring agents. Then, it finds which system has a more load rather than its current load. To save network resources and its bandwidth, Supervisor agent could not reside on all the system in the network. It present only in one system and move to the network. There are three components in this supervisor agent, which are communication, analysis, and response components. It has set of rules and regulation and it analyse whether any suspicious activity has been made in the network or not and gathers an system target information

Communication agent gets communicated with the monitoring agents for gathers an information for find out the load target of the system. Analysis agents finds out the neighboring node which have more target rather than its present load system using existing algorithm . The approach to select a node may be competitive or negotiated. In addition to that, analysis agent uses set of rules and regulation for comparing the predefined data with collected data for security. Then, it communicates with the locate agent to move with all information from present node to the target node and it act as a server and to distribute the work effectively against the load and also if any suspicious activity found, it locate this agent and it disconnect the respective node from the network.

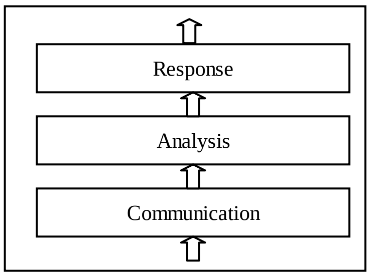

Figure 2 Supervisor Agent

c. Locate Agent

Locate agent is an moving agent to move all the information along with the virtual server to the heavy load system and it act as an virtual server to distribute the work very effectively. There are three components are move, Locate and set up agent which is responsible for locate the virtual server to the heavy load system. Locate components are to find out the system where to locate the virtual server for distribution of the work. Move components are move the virtual server from the present system to heavy load system. Set up agent is to set up the virtual server in that heavy load system for analysis the load balancing factors.

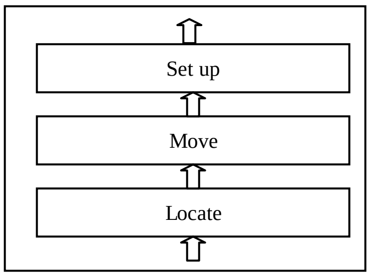

Figure 2 Monitoring Agent

IV. Conclusion

This paper proposed a system to efficiently distribute the loads among the nodes in the network efficiently. Our approach using Mobile agent as a core to collect the information from all the nodes and rearrange the loads efficiently with security. Our mechanism also provides advantages over the other existing techniques whose application requires load balancing system efficiently with security. Traditional mechanism have not been sufficient to provide the distribution of the work efficiently with security, however, this mechanism is able to distribute the work efficiently and to detect known and unknown attacks which is present in this system.

References

[1] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker,"A Scalable Content-Addressable Network," in Proc. ACM SIGCOMM, San Diego, 2001.

[2] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek,and Hari Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications," in Proc. ACM SIGCOMM, San Diego, 2001, pp. 149–160.

[3] Kris Hildrum, John D. Kubatowicz, Satish Rao, and Ben Y. Zhao, "Distributed Object Location in a Dynamic Network," in Proc.ACM SPAA, Aug.2002.

[4] Antony Rowstron and Peter Druschel, "Pastry: Scalable, Distributed Object Location and Routing for Large-scale Peer-to-Peer Systems," in Proc. Middleware, 2001.

[5] Ananth Rao, Karthik Lakshminarayanan, Sonesh Surana, Richard Karp, and Ion Stoica, "Load Balancing in Structured P2PSystems," in Proc. IPTPS, Feb. 2003.

[6] John Byers, Jeffrey Considine, and Michael Mitzenmacher,

[7] "Simple Load Balancing for Distributed Hash Tables," in Proc.IPTPS, Feb. 2003. Frank Dabek, Frans Kaashoek, David Karger, Robert Morris,

[8] and Ion Stoica, "Wide-area Cooperative Storage with CFS," in Proc.ACM SOSP, Banff, Canada, 2001. David Karger, Eric Lehman, Tom Leighton, Matthew Levine,Daniel Lewin, and Rina Panigrahy, "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web," in Proc. ACM STOC, May 1997.

[9] M. Adler, Eran Halperin, R. M. Karp, and V. Vazirani, "A stochastic process on the hypercube with applications to peerto-peer networks," in Proc. STOC, 2003.

[10] David Karger and Matthias Ruhl, "New Algorithms for LoadBalancing in Peer-to-Peer Systems,"Tech. Rep. MIT-LCS-TR-911, MIT LCS, July 2003.

[11] J. R. Douceur and R. P. Wattenhofer, "Competitive Hill-Climbing Strategies for Replica Placement in a Distributed File System,"in Proc. DISC, 2001.