RESULT ANALYSIS OF FAULT TOLERANT PERMISSION BASED CLUSTERED MUTUAL EXCLUSION ALGORITHM IN DISTRIBUTED SYSTEM

Rishikesh Rawat
PhD Research Scholar
Mewar University, Chittorgarh,
Rajasthan, India

Dr. Sarvottam Dixit
Professor
Department of Computer Science & Engineering,
Mewar University ,Chittorgarh
Rajasthan, India

Abstract: Due to the rising application of peer to peer computing, distributed applications are continuously spreading over an extensive number of nodes. Distributed system is a collection of different processes that do not share memory or clock. These processes are connected by a communication network in which each process has its own local memory and other peripherals. The message between any two processes of the system takes place by message passing over the communication network. These processes in the distributed system run concurrently. The Main Aim of the distributed system is to provide an efficient and convenient environment for sharing of resources. we present an Permission-Based clustered Mutual Exclusion Algorithm in distributed systems. This algorithm provides better result than Reddy algorithm. The algorithm is depend on token-based technique and requires a lot number of messages to enter the critical section (CS) and to detect the system failures. In Reddy algorithm the process must request to enter the CS from all process all the times, we used a technique wherein it can enter the CS by sending less number of messages. Also Proposed algorithm reduces the number of messages for the detection of the token lost. We compare our algorithm with Reddy and we find that Proposed algorithm is more efficient.

KEYWORDS: Mutual Exclusion, Permission based algorithm, Fault tolerance, Distributed system

1. INTRODUCTION

Mutual exclusion is one of the first problems met in parallel processing. The implementation of a Mutual Exclusion mechanism is a major task facing every designer of Operating Systems. It is of value to Application Programmers also because they have to use services provided by computer systems built around several processing units, or even several computers linked by a network. The problem of Mutual Exclusion is : to define fundamental operations that make it possible to resolve conflicts resulting from several concurrent processes sharing the resources of a computer system. The whole complex structure of synchronization and communication between the elements of a parallel or distributed system depends upon the existence of such operations. In a Centralized system, all processes share a common memory. Therefore, the synchronization of the accesses to shared data may be achieved through simple primitives operating on a common integer variable called semaphore. A process that needs shared data executes a test on the value of the semaphore (state of the shared resources) to see if it is greater than zero. If so, it decrements the value and continues. Upon finishing the use of the shared data, the process increments the value of the semaphore. If the value of the semaphore is zero, the process is put to sleep and will be woken up when the semaphore's value is greater than zero. Hence, the correctness of the protocol for the management of the accesses to shared data, the ordering of the primitives and a centralized control are necessary for the correctness of the algorithm (neither deadlock nor starvation). In distributed system many problems including concurrency control of replicated data, atomic commitment, distributed shared memory and others require the execution of operations in a mutually exclusive way. Unfortunately, in distributed systems, neither shared memory nor a common clock is available. Therefore, the protocol for the management of the accesses to a shared resource will be more difficult since the ordering of the activities must be done by means of communication messages whose delays are unpredictable. Two types of control are possible for distributed mutual exclusion algorithms:

• Centralized control : In this the state of the resource is always in the private data space of one co-ordinator process and each process sends requests to the coordinator and waits for a grant.

• Distributed control: In this the• state of the shared resources is replicated in the private data space of all or some processes, or it is exchanged among the processes. In both the cases, the decision of the allocation of the resources is an agreement among more than one process. One of the main advantages of the distributed control is that it is resilient to failures. The centralized control is not. Moreover, centralized control needs two different process codes, one for the coordinator and one for the requesting process. This complicates recovery after failure. The drawback of the distributed control is more complex.

2. RELATED WORK

Sahel Alouneh, et al.,[1] has proposed and focused on path protection fault tolerance schemes as well as issues in MPLS (Multi-protocol label switching) network and steps to resolve it. MPLS networks are prone to failures thus fault tolerance is important as it focuses on factors like utilization, recovery time and packet loss. MLPS provide VPN functionality to increase security. The advantages of this method are high speed packet delivery, reduction in PSL (Path switching LSR), fast and cost effective notification. But packet loss factor depends upon the various approaches without any guarantee from VPN and it is the disadvantage of this method. Further works needs to be data integrity, confidentiality, authentication in MPLS and multi path routing.

Moushumi Sharmin,et al., [2] has proposed to design a resource discovery unit used to maintain privacy, resource sharing and modification scheme in an effective computing environment. Fault tolerance techniques uses secret sharing in which the process will not keep a single copy of resource manager to ensure the optimum use of tiny storage. For the security & reliability resource provider provides privacy and security to a specified resource and resource holder is responsible for providing it to approved person only. Resource manager manages resources, facilitates the best match and stores the address in the hash table. Disadvantages are in this increases seek time, wastage of resources & memory, low privacy of service information and domain identity.

M.AI-Kuwaiti et.al. [3] has proposed address issues of complex infrastructure such as information, reliability and availability with some fault tolerance and security features. In fault tolerance the system behaves normally in case of any hardware or software fault and fault masking is another method that can be used to tolerate fault. Reliability refers to any failure free operation during an interval and security protects confidentiality & integrity..

Ben Hardekopf,et al., [4] has proposed a system with security and fault tolerance in case of congestion and protocols to create a secure voting algorithm. Durability & Redundancy is used to reduce the risk of any single component in operating flawlessly is the advantage of this approach. Future work is that reduce network congestion, reliability & security for the WAN.

Jitendra K. Rout, et.al.,[5] has proposed a fault tolerance paradigm that can identify the black holes attack which degrades the performance of the network by dropping the packets. In the fault tolerance dividing the routing protocols into zones, if an error occurs in one then the other zones does not get effected and a node connection algorithm is used to detect where the fault has actually occurred. For the security purpose network connection algorithm used for fault tolerance and provide reliability. Its advantages are that many routing protocols have been discovered because of which data transmission and maintenance has been enhanced. Due to this increase in the number of routers, bandwidth and cost will be effective

Steven E.Crerwinski et.al. [6] has proposed to handle the failure automatically by hiding the complexity of fault recovery and to determine whether the communication among the components is secure or not. For the fault tolerance whenever a packet gets dropped, cryptographic methods provide strong authentication at end points. SDS server is used to calculate the number of clients in the system and verify that the security features of the system does not reduced and Authenticated RMI implementation is used to encrypt the data sent over the network. Its advantages are that it scalable, secure information repository and fault tolerant. Lack of access control, service information cannot be granted and cost in effective

Yaron Minsky et al., [7] has proposed a variety of applications in the internet and other large distributed systems. Replication and voting are not sufficient for improving the performance of agent computation. It involves fault tolerance, if there exists no faulty host in the pipeline then the remaining hosts separate the computations by sending an agent. The correctness of the present state depends upon the correctness of predecessor. It uses chain of authentication to prevent masquerading and when two hosts combine then voting will determine which faulty host is encountered and which is not, it is used for security factor .

Stephen Bohacek et.al.,[8] has proposed simulation that improves routing security, minimize the impact of link router and approaches to failure prevention and recovery. Used dynamic routing protocol to try to reduce the failure, when a fault is generated in one layer then transmission will keep following the same route and continue interception. Security contains protest against some form of interception; provide end to end security mechanism to other layers. Its advantages are to improve security and fault tolerance; proactive approaches to achieve connectionless value enable failure. Demerits are increasing throughput and complex multiple parts. For future work, scalable algorithm is required to compute next hop probabilities, decrease packet transmission delay and increase throughput.

Bharat B. Madan,et.al., [9] has proposed assessment of security attributes for an intrusion tolerates system. Fault tolerance ensures the effective recovery from failures and allows a finite probability; the system security may be breached. SMP model deals with security, Integrity and authorized actions. Merits are it is able to detect faults in the system and detect the insertion of the security attacks into a system. Various subsystems communicate with each other. But it can't compromise with data integrity and DOS attacks, consume large amount of service resources..

Kevin J.Rowett [10] has proposed to various fault tolerance designs such as redundancy part, data checking and multiport transmission. In the fault tolerance the pair of network control device sends a message to the provider of the other pair of the network and identifies the other pair of network which operates continuously. It is for secure & reliable hub to hub connection to transmit data and provide fault redundant network..

David Tipper et.al.,[11] has proposed survivability framework and focuses on voice service network. It generates wireless access network and reliability. When fault occurs user provides continuous service and non-user cannot access the system without authentication. Mitigate the impact of failures. Cost issue, long distance, range problem and no network are the main problems in this approach. It implements in Checkpoint algorithm. It is for the future perspective to extend the scope of services.

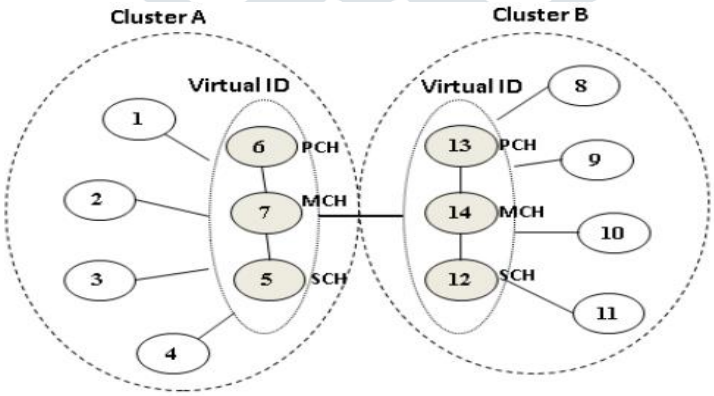
3. Problem Identification

Distributed system is a collection of distinct processes that do not share memory or clock. These processes interconnected by a communication network in which each process has its own local memory and other peripherals. The communication between any two processes of the system takes place by message passing over the communication network. These processes in the distributed system run concurrently. The purpose of the distributed system is to provide an efficient and convenient environment for sharing of resources. In this paper, we present an efficient fault-tolerance algorithm for Mutual Exclusion (ME) in distributed systems. This algorithm is considered as an enhancement to Reddy algorithm. The algorithm is based in token-based technique and requires a lot number of messages to enter the critical section (CS) and to detect the system failures. In Reddy algorithm the process must request to enter the CS from all process all the times. The presented algorithms also can be seen from another point of view, the fault tolerant; we said that the algorithm is fault tolerant if it treats the failure and still function without any problems. There are many algorithms that take into account several types of failures and treat them [8]. By contrast, we cannot consider that the system is reliable and it is error free so to build an algorithm over this suggestion is not practical and they cannot be classified with the others that take in their accounts the error and any failures in the system. The fault tolerant algorithms could be different in their efficiency and complexity. For example, in token-based algorithm the efficiency depends on the number of messages needed for entering the CS, for detecting loss of token, and for generate new token.We present in this thesis an efficient Permission-Based clustered Mutual Exclusion Algorithm that reduces the number of messaged needed to enter the CS, the number of messages to detect the loss of token and to generate a new token in case of failures

4. AN EFFICIENT ALGORITHM FOR DISTRIBUTED MUTUAL EXCLUSION IN DISTRIBUTED SYSTEM

4.1. Proposed Technique

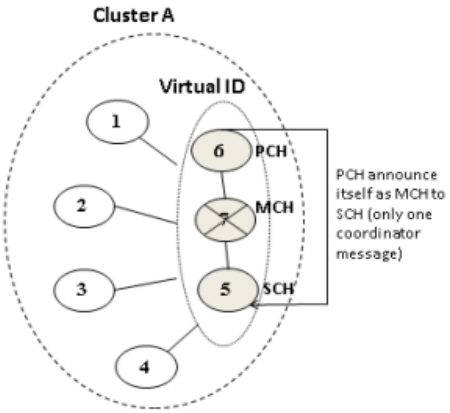
The prime motivation of our proposed algorithm is to guarantee that the new cluster head can be elected with less number of messages, less number of participants in the election process and also reduce the single point of failure. The proposed fault tolerant permission based clustered mutex algorithm comprises virtual cluster head election algorithm to tolerate cluster head failure. The set up consists of a Main Cluster Head (MCH), two backup heads namely Primary Cluster Head (PCH) and Secondary Cluster Head (SCH). Node with highest ID is selected as MCH and the next two highest ID nodes within the cluster will be selected as PCH, SCH. The main cluster head coordinate the cluster members i.e., handle the CS request and primary, secondary cluster heads observe the main cluster head. A virtual ID is given to cluster heads, the members will send CS request message to this ID and initially virtual ID is mapped to main cluster head. The cluster members are not aware of this virtual group.



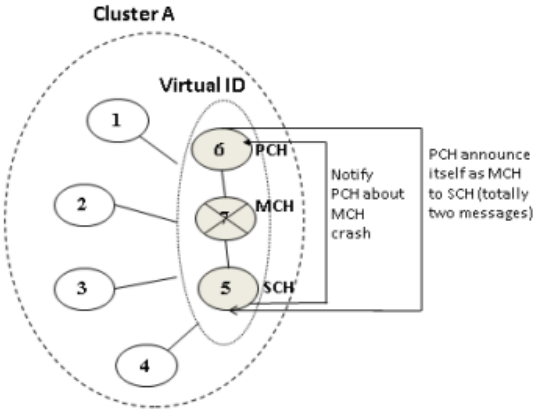
Fig(1): Virtual Cluster Head Election Algorithm

Fig.1 shows the model of virtual cluster head election algorithm. Nodes are first grouped into cluster then cluster heads are selected. For cluster A, the highest ID node is 7 and it is selected as Main Cluster Head, next highest ID 6 is selected as Primary Cluster Head and next ID 5 is chosen as Secondary Cluster Head. A virtual ID is created and it is mapped to MCH. Node 5 and 6 continuously monitor node 7. The bold line between two clusters indicates the connection between cluster heads i.e., cluster A‟s virtual head ID to cluster B virtual head ID. The primary and secondary cluster head constantly monitor the main cluster head by sending „Are You Alive (AYA)‟ message periodically and the MCH responds by sending „Yes I am Alive (YIA)‟ message. If

there is no response from MCH (best case), primary will take over the job of main cluster head, map its ID to virtual ID and intimate to secondary cluster head as a new head and it is represented in Fig.3. If the failure is detected by secondary cluster head (worst case) it will intimate to primary cluster head about main cluster head and the primary will take over further operation, denoted in Fig.4. If primary doesn't respond properly, the secondary cluster head will coordinate the members. In this way the election is done by using only three nodes in the cluster. This will reduce the participant "number in the election process and also the number of messages needed for declaration of a new cluster head.



Fig(3): Failure of Main Cluster Head noticed by Primary Cluster Head



Fig(3): Failure of Main Cluster Head noticed by Primary Cluster Head

5. Performance Analysis

Number of messages per CS entry (MPCS):

The algorithm says that the average number of messages to enter into the CS is incoming link set and out gong link set is equal. So the number of messages exchanged among the hosts for each execution of the CS. Therefore the MPCS under low load condition is ;

$$MPCSlow = (n + n) / 2 = n$$

Under the high load condition n/2 hosts issues current request messages .Each request message the reply message must be send so half of these hosts are in outgoing link list is n/4. Therefore the MPCS under the high load condition is:

$$MPCShigh = (n/2 + n/4) = 3 * n/4 .$$

Table.1: The table shows #MPCS the high and low load

No. of Hosts	MPCS-Low	MPCS-High
0	0	0
20	20	7.5
40	40	1.5
60	60	22.5
80	80	37.5

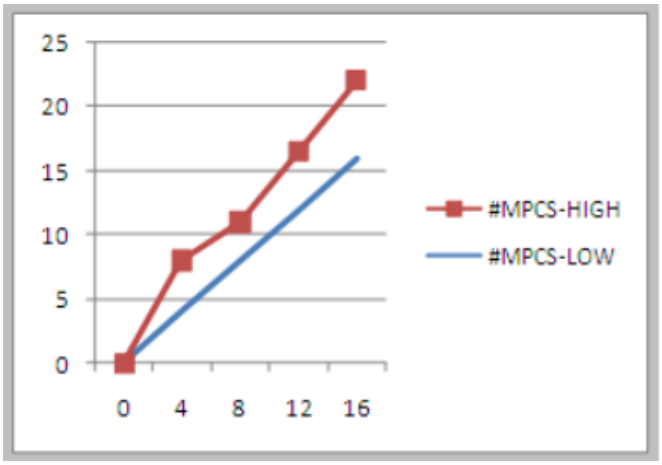


Fig.2: The graph shows the #MPCS at high and low load

No. of Hosts	MPCS-Low	MPCS-High
0	0	0
4	4	4
8	8	3
12	12	3.5
16	16	6

Table7.2: The table shows #MPCS the high and low load

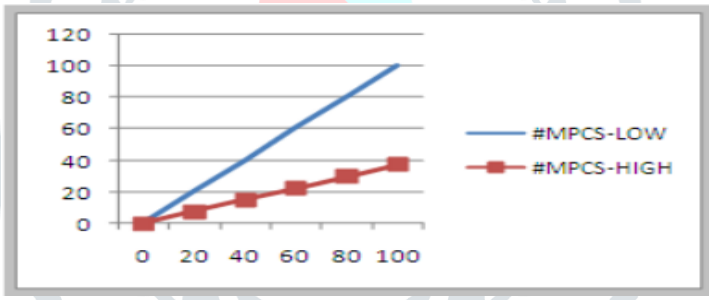


Fig. 3: The graph shows the #MPCS at high and low load

Synchronization delay (SD): The number of sequential messages exchanged after a host leaves the CS and before the next host enters the CS .Under low load conditions the synchronization delay is meaningless, because it measures the interval between the arrival of two requests. Under the high load conditions, when host Si exits the CS, it will send REPLY messages to all the hosts in its Incoming_Lset, i.e. that is the host that have pending requests at the earliest time will enter the critical section immediately after it receives the REPLY from Si . Therefore, the Synchronization delay under the high load condition is

SDhigh=1, i, e.one message transferring time.

Response time: The time interval that a host waits to enter the CS after its request for CS arrives. Under low-load conditions, most of the time, no more than one host competes for the CS. When a host wants to enter CS, it sends REQUEST messages to the hosts in its Incoming_Lset and then all these hosts send REPLY immediately after they receive the REQUEST. Therefore the response time under the low-load level is: RTlow = 2, i.e. twice of the time of transferring one message. Under high-load conditions, there is always a pending request at each host. The hosts are in the waiting queue respect to the timestamps of their requests, i.e. the time when they issue requests for CS. A host in the queue can enter CS after its predecessor exits, so each host needs to wait for the hosts whose requests are earlier. On average, each multitude has to wait for n/2 such hosts. supercilious the average time of an running of CS is A, the response time under high-load conditions is

RThigh=(A+SDhigh)*n/2=(A+1)*n/2

Average Time	No. of Host	RThigh
2	5	7.5
3	10	20
4	15	37.5
5	20	60

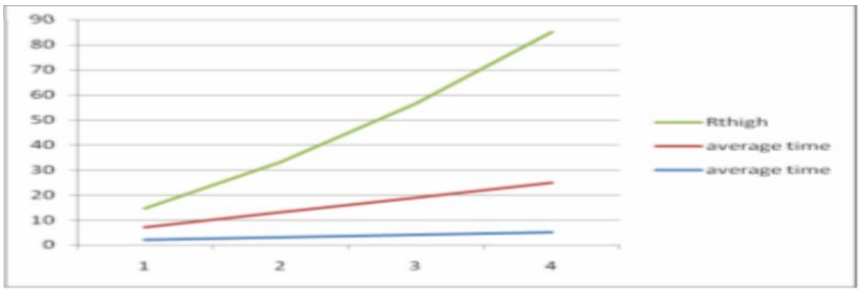


Fig 4: The graph shows the RThigh at different average time and different no of hosts

6.CONCLUSION

We proposed a new fault-tolerant distributed mutual exclusion algorithm in distributed system which assumes a fault-free system. To detect process failures, we used acknowledgements and timeout based re-transmission of messages which are commonly used techniques for fault detection in a distributed system.

We present suitable algorithm to solve the problem of mutual exclusion in fully connected network. Our algorithm enhances the algorithm of Reddy et al's performance. Proposed algorithm can operate in reliable system and it can tolerate the node failures. Also we presented an suitable fault-tolerant token-based mutual exclusion algorithm for distributed system which is an extension of Reddy et al's algorithm and made main adjustments that we have got proposed algorithm to improve performance by reducing number of messages that necessary to detect token loss and to generate token. Similarly, by reducing number of rounds of communications. As well as we obtained reducing synchronization delay as compared to, which in turn leads to lower response time. s. s. Most fault-tolerant token-based mutual exclusion algorithms try to generate a new token by a technique similar to leader election, in the event of token loss. We used a novel technique to find the site which executed the CS most recently and use the information contained in that site to generate a new token. Our token generation process is more dynamic and distributed in nature. In this thesis In this paper we compared between two mutual exclusion algorithms from different aspects and norms such as number of messages per CS execution, Scheduling of processes to enter critical section and dealing with some failure cases. Also we presented an suitable Permission-Based clustered Mutual Exclusion Algorithm for distributed system which is an extension of Reddy et al's algorithm and made main adjustments that we have got proposed algorithm to improve performance by reducing number of messages that necessary y to detect token loss and to generate token. Similarly, by reducing number of rounds of communications. we received reducing synchronization delay as compared to, which in turn leads to lower response time.

REFRENCES

1. Sahel Alouneh, Sa'ed Abed "Fault Tolerance and Security Issues in MPLS Networks" pub. ISSN: 1792-4863 ISBN: 978-960-474- 231-8.
2. Moushumi Sharmin, Shameem Ahmed, and Sheikh I. Ahamed " SAFE-RD (Secure, Adaptive, Fault Tolerant, and Efficient Resource Discovery) in Pervasive Computing Environments".
3. M. Al-Kuwaiti, N. Kyriakopoulos "A Comparative Analysis of Network Dependability,Fault-tolerance, Reliability, Security, and Survivability" pub.in IEEE communication surveys & tutorials, VOL. 11, NO. 2, SECOND QUARTER 2009.
4. Ban Hardekopf,Kevin Kwiat"Secure and Fault –Tolerant Voting in Distributed Systems" published by 0-7803-6599-2/01/$10.00 _ c 2001 IEEE.
5. Jitendra Kumar Rout, Sourav Kumar Bhoi "SFTP: A Secure and Fault-Tolerant Paradigm against Blackhole Attack in MANET" pub.in International Journal of Computer Applications (0975 – 8887) Volume 64– No.4, February 2013.
6. Steven E. Czerwinski, Ben Y. Zhao "An Architecture for a Secure Service Discovery Service" pub. In Mobicom '99 Seattle Washington USA Copyright ACM 1999 I-58113-142-9/99/08.
7. Yaron Minsky, Robbat Van Revesse "cryptographic support for fault tolerant distributing computing".
8. Stephan Bohacek, Jo.ao Hespanha" Game Theoretic Stochastic Routing for Fault Tolerance and Security in Computer Networks".

9. Bharat B. Madan, Katerina Goševa-Popstojanova "A method for modeling and quantifying the security attributes of intrusion tolerant systems" pub.in 0166-5316/$ – see front matter © 2003 Published by Elsevier B.V.
10. Kelvin J.Rowett "Method and apparatus for fault tolerance connection of a computing system to local area network", published by US005448723A Pub.Date: Sep. 5, 1995.
11. David Tipper, Teresa Dahlberg "Providing Fault Tolerance in Wireless Access Networks" pub.in IEEE Communications Magazine • January 2002.