*International Journal of Scientific Engineering and Applied Science (IJSEAS) – Volume-2, Issue-1, January 2016*
*ISSN: 2395-3470*
*www.ijseas.com*

# Integrity Improvement Using Controlled Data Replication in Distributed Environment

**Sonali Gwalherkar[1], Lalit Gehlod[2]**
[1] Institute of Engineering and Technology DAVV, Indore, India
[2] Institute of Engineering and Technology DAVV, Indore, India

## Abstract

Replication is a strategy for improving reliability, fault tolerance and availability in distributed systems. Replication can be defined as keeping several copies or replicas of the same file at different nodes. This reduces server load, access latency and network congestion. One of the most important factors that affect replication is integrity of replicas. So here purpose is to implement an algorithm to detect integrity of replica for distributed environment. It provides fault tolerance while running on inexpensive commodity hardware. This makes it possible for multiple clients on multiple machines to access files and storage resources transparently. While sharing many of the same goals as previous replication models, objective is to improve integrity using controlled data replication in distributed environment.

***Keywords:*** *Replication, Concurrency, integrity, fault tolerance, consistency*

## 1. Introduction

A distributed system is collection of computer nodes or devices that are geographically distributed and connected through a communication link [4]. Example of distributed system considers inventory control system, banking system, airline reservation etc. Here, we propose a technique for ensuring secure replication in distributed systems. Proposed solution will give special security features whereas first one is confidentiality and another one is integrity of replicated files. A distributed system is a piece of software that ensures that: A collection of independent computers that appears to its users as a single coherent system. There are two aspects of distributed system; (1) independent computers and (2) single system or middleware. Tanenbaum & van Steen define that a distributed system is a collection of independent computers that appears to its users as a single coherent system. A distributed system organized as middleware. Layer of software offering a single-system view offers portability and interoperability simplifies development of distributed applications and services. It is shown in figure 1.1.

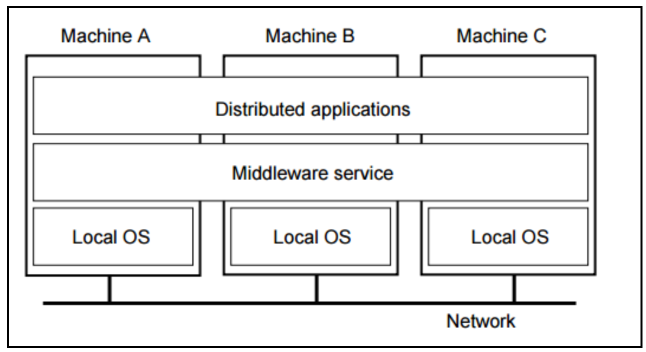


Figure 1.1: Block Representation of Distributed System

## 2. Behavior of Distributed Systems

A distributed system has several advantages over single machine in context of throughput, efficiency. Following characteristics define a distributed system behavior:

### 2.1 Concurrency

Concurrency is obtained by parallel processing of several parallel nodes operating together.

### 2.2 No global Clock

It makes coordination difficult for ordering of events

### 2.3 Independent Failure of Components

The complete phenomena conclude that "partial failure" & incomplete information may result.

## 2.4 Unreliable and Unsecure Communication

Due to weak security measures and possibility of un-authorized access and recording may lead to leakage of security Loss of connection and messages. There may be possibility of possibility of unauthorized recording and modification of messages.

## 2.5 Expensive Communication

Communication between computers usually has less bandwidth, longer, latency, and costs.

## 2.6 Openness and Extensibility

Interfaces should be cleanly separated and publicly available to enable easy extensions to existing components and add new components.

## 2.7 Migration and Load Balancing

Allow the movement of tasks within a system without affecting the operation of users or applications, and distribute load among available resources for improving performance.

## 2.8 Security

Access to resources should be secured to ensure only known users are able to perform allowed operations.

# 3. Transparency of Distributed Systems

Distribution transparency may be set as a goal, but achieving it is a different story. Here, list of distributed system transparency has been written below:

## 3.1 Access:

Hides the access by not letting to whether the accessed document is replica or main file.

## 3.2 Location:

Hides where an object resides

## 3.3 Migration:

The ability of a system to change that object's location is hidden from object.

# 4. Replication

Replication is the process of creating and managing duplicate versions of a database or file or content to maintain backup or duplicate copy of original one. Replication not only copies a database but also synchronizes a set of replicas so that changes made to one replica are reflected in all the others. [2] The beauty of replication is that it enables many users to work with their own local copy of a file but have the file updated as if they were working on a single, centralized database. For file applications where users are geographically widely distributed, replication is often the most efficient method of database access. Two replication strategies have been used in distributed systems: Active and Passive replication. In active replication each client request is processed by all the servers. The big disadvantage for active replication is that in practice most of the real world servers are non-deterministic. Still active replication is the preferable choice when dealing with real time systems that require quick response even under the presence of faults or with systems that must handle byzantine faults. In passive replication there is only one server (called primary) that processes client requests. The disadvantage of passive replication compared to active is that in case of failure the response is delayed.[2] Figure 4.1 shows the replication method of active and passive replication.

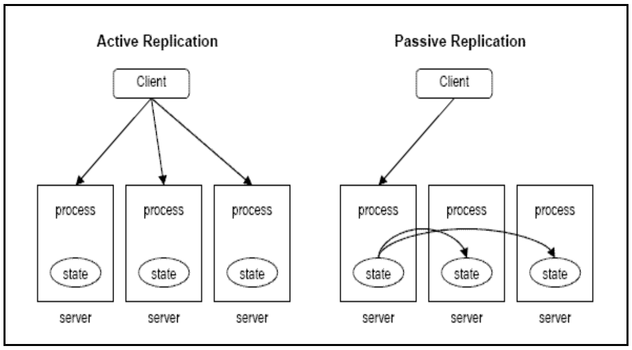

Figure 4.1: Active and Passive Replication

There are many examples of replication schemes in distributed file and database systems. Replication Protocol used in [6] uses a primary-copy with server redirection scheme that offers strict or sequential consistency without imposing any overhead on normal reads.

## 5. Challenges for Replication

Replicating data at many nodes and allowing anyone to update the data is problematic. Security and performance both have tradeoffs. Eager replication keeps all replicas exactly synchronized at all nodes. Eager replication gives serialized execution – there are no concurrency anomalies. But, eager replication reduces update performance and increases transaction response times because extra updates and messages are added to the transaction. Mobile applications require lazy replication algorithms. These algorithms asynchronously propagate replica updates to other nodes after the updating transaction commits.[8] Following challenges are observed in replicated environment which may be listed below:

### 5.1 Data Consistency

Maintaining data integrity and consistency in a replicated environment is of prime importance. High precision applications may require strict consistency of the updates made by transactions. Consistency, accessibility, scalability, security, fault tolerant and performance are areas for system implementation. High accessibility and performance are basics for such a system with large-scale distributed database system. We have to make a tradeoff between consistency and replication. There are dissimilar levels of weak and strong consistency. Also there are levels of consistency such as data-centric and client-centric models for a data replication mechanism. So recognizing usage of consistency models in each data replication mechanism is necessary. [3]

### 5.2 Downtime during new replica creation:

If strict data consistency is to be maintained, performance is severely affected if a new replica is to be created. Sites will not be able to fulfill request due to consistency requirements.

### 5.3 Maintenance overhead:

If the files are replicated at more than one site, it occupies storage space and it has to be administered. Thus, there are overheads in storing multiple files.

### 5.4 Failure:

Hierarchy of failures in distributed systems, Omission failure occurs when a component fails to respond to a service request. Typical omission failures include server crash or communication link outage. Performance failure occurs when a system component fails to respond to a service request within the time limit specified for the delivery of that service. Occasional message delays caused by overloaded servers or network congestion are examples of performance faults. In Byzantine failure, components act in arbitrary, even malicious ways. Compromised security can lead to Byzantine failure. [6]

### 5.5 Lower write performance:

Performance of write operations can be dramatically lower in applications requiring high updates in replicated environment, because the transaction may need to update multiple copies.

## 6. Problem Definition

In centralized systems all data are stored at server. In case of heavy data access traffic at server increases resulting in reduced data availability at the client side whereas in distributed systems the problem is less severe. There are three criteria for replication that affects the behavior of distributed system first is number of replication copies needed, second best location for replica before creating a replica [5]. Data availability has been improved through replication of data at multiple servers. The existence of multiple copies of same data alleviates the traffic problem of centralized systems, but this data availability has been achieved at the cost of inconsistency. The complete work observes that replication and integrity are two most crucial issues in distributed system. Furthermore, the work also examine that security is also major issue in replication process. Un-authorized user may attempt to compromise the content of replica files or content of original file. There may be possibility of modification of content in replication file. The complete work observes that there is big need to develop a security solution in association with replication method to efficient access with sufficient security.

## 7. Solution Domain

A distributed environment raises a number of security issues. First, the broadcast nature of most local area networks makes them particularly vulnerable to eavesdropping [1]. Anyone with a personal workstation on an Ethernet can easily monitor all network traffic. Secondly, the lack of control over the software run in an individual workstation makes masquerades, replays, and similar active threats possible. These problems are solved in single-machine or centralized environments by physical security: locked machine rooms and protected terminal lines. Unfortunately, the decentralized nature of distributed systems precludes such measures. Logical rather than physical schemes must be used instead. The simplest problem to solve is that of eavesdropping.
The solution uses encryption: two persons wishing to communicate do so by encrypting all their messages with a secret key known only to them. This effectively constructs a secure private communication channel on top of the underlying insecure public channel. Therefore, we need a security mechanism [7] that enforces the following properties on the replicated state of a data:
**Authenticity** - the document the client receives from a server has indeed been created by the object's owner. No attacker or malicious server should be able to pass off one of their own documents as being part of the object.[7]
**Freshness** - the client is guaranteed to receive the most recent version of a document part of an object. No attacker or malicious server should be able to pass off genuine but old versions of a document and convince the client they are fresh.
**Integrity** - the client is guaranteed to receive a document, part of the object that is correct and valid. No attacker or malicious server should be able to modify the requested document with incorrect data.

My proposed work direction in the related area:

1. Replication is necessary and cannot be avoided. But the manner of replication must be defined so that the location of next replica is a default location and is known.

2. Defining the manner of replication will avoid the need of object locating mechanism.

3. The location of original files and their owner description must be kept at a repository server.

4. Reduction in network traffic at server side and optimizing bandwidth utilization.

5. Improving data availability and access.

6. Maintaining consistency keeping the above points in mind.

7. SHA-1 algorithm will help to maintain integrity of content and security among replicas.

## 8. Conclusions

Despite continues effort for file replication and consistency management, a secure method is still required. Proposed solution helps to implement secure and integrity improved controlled replication in distributed system.

## References

[1]Danny Dolev, Andrew c. Yao, "On the Security of Public Key Protocols", IEEE Transactions on information theory, VOL. IT-29, NO. 2, MARCH 1983

[2] Sanjay Kumar Tiwari, A K Sharma and Vishnu Swaroop, "Issues in Replicated data for Distributed Real-Time Database Systems", International Journal of Computer Science and Information Technologies, Vol. 2 (4) , 2011, 1364-1371

[3] Alireza Souri, Saeid Pashazadeh and Ahmad Habibizad Navin, "Consistency of Data Replication Protocols in Database Systems a Review", International Journal on Information Theory (IJIT), Vol. 3, No. 4, October 2014.

[4] Maurice P. Herlihy, J. D. Tygar, "How to Make Replicated Data Secure", Carnegie Mellon University Computer Science Technical Report, CMU-CS-87-143, August 1987.

[5] Bakhta Meroufel, Ghalem Belalem, "Managing Data Replication and Placement Based on Availability", AASRI Conference on Parallel and Distributed Computing Systems, 2013, 147 – 155

[6] Jiaying Zhang, Peter Honeyman, "Replication Control in Distributed File Systems", CITI Technical Report 04-01, April 2004.

[7] Bogdan C. Popescu, Janek Sacha, Maarten van Steen, Bruno Crispo, Andrew S. Tanenbaum, Ihor Kuz, "Securely Replicated Web Documents", Vrije Universiteit, Amsterdam, The Netherlands

[8] Jim Gray, Pat Helland, Patrick O'Neil, Dennis Shasha, "The Dangers of Replications and a Solution", Vrije University, Amsterdam, the Netherlands

**Sonali Gwalherkar:** Bachelor of Engineering in Computer Science in 2010, currently pursuing Master of Engineering specialization in software engineering; Worked as Lecturer in Bansal Group of institutes Indore, current research interests are operating system and computer architecture.

**Lalit Gehlod:** Bachelor of Engineering in Computer Science, M.E in Computer Science and Working as a Lecturer in Institute of Engineering DAVV Indore Madhya Pradesh India.