Resilience and Fault Tolerance in Cloud Computing

Arun Kumar S[1], Gayathri[2], Anusha M S[3], Uttam Upendra Hegde[4], Vinay Kumar G R[5,] N Nauman Pasha[6]

[1]*Assistant Professor, Department of Computer Science & Engineering, Presidency University, Bengaluru, Karnataka, India*

[2,3,4,5,6]*Student, Department of Computer Science & Engineering, Presidency University, Bengaluru, Karnataka, India*

***Abstract*— The rising need for adaptable and scalable computing resources has led to the widespread adoption of Cloud computing, offering convenient access to resources without the hassle of hardware ownership or software maintenance. However, despite its benefits, the Cloud model is vulnerable to a variety of system failures, creating customer worries about service reliability and availability. Resilience and Fault tolerance mechanisms aim to alleviate these worries by ensuring system stability and continuity.**

***Index Terms*— Fault Tolerance, Cloud Computing, Fault Model, Layered Architecture**

I. INTRODUCTION

Cloud Computing involves accessing computer resources like data storage and processing power as needed, without direct user control over physical infrastructure. Users typically aren't fully aware of how the cloud operates, where their data is stored, or how it remains accessible at all times. Often, large cloud services are distributed across multiple data centers, promoting resource sharing and often adopting a pay-as-you-go payment model. Since around 2010, Cloud computing is gaining traction over traditional computing platforms [1]. service providers are creating large data centers distributed over numerous geographic regions. Data centers are meant to house thousands of servers, virtualize and distribute computing resources (such as CPU processing power, storage , and RAM) across the internet, often on a pay-per-use basis.

This cost-effective approach eliminates the need for individual physical hosts by transforming them into a pool of virtual hosts or machines, always available to deliver computing resources and other cloud services when required by all the users in the world. The output as such is that many customers/users are using cloud services as a better option [2]. Millions of users worldwide rely on cloud services, including various organizations, but even meticulously designed data centers can face unforeseen challenges beyond their original scope due to the high load. The intricate infrastructure of these data centers renders them susceptible to various failures, impacting the dependability and accessibility of cloud computing services.

These challenges underscore the need for strong fault tolerance methods to mitigate the impact of potential disruptions on cloud services. By implementing proactive strategies, both users and service providers can uphold the integrity and continuity of system operations, fostering trust and reliability in cloud computing environments. Thus, fault tolerance becomes critical for both customers and service providers, ensuring that system operations continue despite an unanticipated array of faults.

This chapter aims to explore the characteristics, frequency, and varieties of faults encountered in typical Cloud computing setups, their effects on user applications, and strategies for effectively and economically managing these faults.

First, we describe the fault model seen in most Cloud computing configurations, taking into account the system architecture, common server and network component failure patterns, and analytical models. We introduce fundamental concepts of fault tolerance and highlight key parameters essential for constructing resilient systems. Following this, we propose a solution capable of seamlessly addressing one of the two primary fault classes prevalent in Cloud computing environments, offering a general and transparent approach for user applications. Furthermore, we explain about a technique that can handle these failures and at a reduced cost which is half compared to existing solutions.

II. FAULT MODEL FOR CLOUD COMPUTING

In essence, a failure occurs when the system fails to meet its intended functionality or expected behavior.

Such failures stem from errors, indicating an invalid system state. Errors, in turn, are often caused by faults, representing fundamental flaws within the system. This sequence of faults, errors, and failures forms the basis for understanding fault tolerance, which ensures system functionality despite encountering failures [3]. Therefore, it's crucial to precisely define correct system behavior to facilitate the development of fault-tolerant systems. In order to understand the frequency and reasons of recurrent system failures, this section examines the fault model of typical Cloud computing settings. The common architecture of CC in order to examine fault distribution and impact.

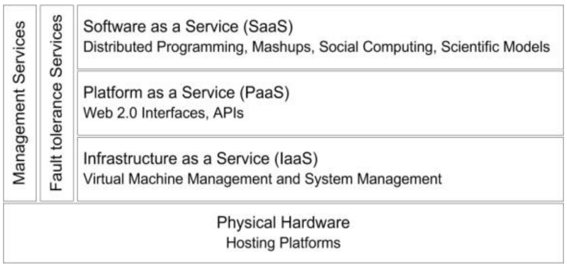

Management Services
Fault tolerance Services
Software as a Service (SaaS)
Distributed Programming, Mashups, Social Computing, Scientific Models
Platform as a Service (PaaS)
Web 2.0 Interfaces, APIs
Infrastructure as a Service (IaaS)
Virtual Machine Management and System Management
Physical Hardware
Hosting Platforms

Fig 1. Layered Architecture of cloud computing

Failure behavior of servers

The investigation into server failures and hardware repair patterns requires access to a substantial array of servers, each equipped with multiple processors, storage disks, memory modules, and network interfaces. This study entails examining a vast dataset encompassing approximately 100,000 servers, along with associated information on part replacements, such as server configurations and timelines for issuing and implementing replacements, including hard disks. This extensive data repository, spanning servers across various data centers situated in different countries, was compiled and analyzed in VN. Observations are:

- Out of all machines, 92% undergo no repair events. However, among the remaining 8%, each machine experiences an average of 2 repairs. This data is based on 20 repair or replacement events identified across 9 machines over a span of 1 year and 2 months.
- A notable 13% of replacements were attributed to a combination of components, lacking a single dominant component failure, while the remaining was due to the functionality of the hard drives.
- Approximately, 5% of servers encounter disk failures within the first year of being commissioned, categorized as “young servers”. This rate increases to 12% when the servers reach one year old, and further to 25% when it reaches 2 year old.
- Server age, configuration and computing capabilities was not found to be the severe reason for the failures of the systems using the Chi squared method.

III. OVERVIEW OF CONCEPTS IN FAULT TOLERANCE

Conceptual Framework of Fault Tolerance

Fault Tolerance (FT) is a strategy for mitigating failures. In light of increasing failures, enhancing fault tolerance is crucial. The intent of fault tolerance is to fortify the dependability and robustness of any system.

The merits of applying fault tolerance in cloud computing encompass recovery from failures, cost reduction, and improved performance metrics. Implementing fault tolerance is triggered when one server for an application which may have multiple instances on various virtual machines crashes.

By adhering to fault tolerance policies and processes, methods are classified into two main groups: proactive and reactive. Proactive fault tolerance anticipates fault retrieval through predictive identification of anomalies before they occur, replacing them with accurate data.

Reactive fault tolerance aims to diminish failures after their occurrence, involving error processing and fault mitigation techniques. Error processing eliminates errors in computations, and fault treatment aims to prevent errors from resurfacing 1997 [4].

Cloud computing service's failure independence is contingent upon its infrastructure architecture, as per . Assuming individual resource failure independence and also considering the location of applications replicas, it is possible to balance resource costs and fault tolerance.

For instance, if aggregate switches malfunction within a cluster, it severs communication among servers. To avert a complete application failure, should such a scenario unfold, locating replicas across clusters becomes necessary. This dictates the relevance of deployment locations in executing fault tolerance mechanisms. In the Cloud computing infrastructure, various deployment scenarios, advantages, and drawbacks are deliberated 2010 [5].

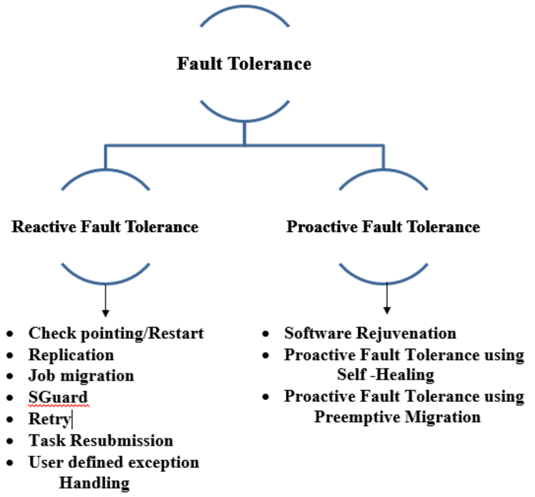

Fault Tolerance
Reactive Fault Tolerance
Proactive Fault Tolerance
Check pointing/Restart
Replication
Job migration
SGuard
Retry
Task Resubmission
User defined exception Handling
Software Rejuvenation
Proactive Fault Tolerance using Self -Healing
Proactive Fault Tolerance using Preemptive Migration

Fig 2. Types of fault tolerance

IV. VARIOUS TIERS OF FAULT TOLERANCE EXIST WITHIN CLOUD COMPUTING

Table 1 depicts the overall availability of different replication schemes across varying deployment scenarios. In fact, traditional methods call for tailored application solutions based on comprehensive infrastructure knowledge, Cloud computing necessitates generic methodologies to broadly apply fault tolerance protocols across user applications.

Agility in managing replicas and enhanced performance should be balanced with cost reduction strategies to optimize resource expenditure without compromising fault tolerance efficacy 2011 [6]. Table 1 shows the availability values for replication strategies for various deployment situations (normalized to 1).

As outlined in Section 3, effectively implementing fault tolerance mechanisms necessitates weighing factors such as the robustness of the fault tulerance model, resoource expenditure, and perfourmance considerations. Traditional methods, which demand custom tailoring for individual applications and extensive knowledge of the used infrastructure, in the context of CC, there's a benefit in devising approaches that can universally apply to user applications. This would enable safeguarding a wide array of applications using a standardized protocol.

Additionally, alongside universality, flexibility in managing replicas and checkpoints is crucial for enhancing performance, while concurrently reducing resource consumption costs without compromising the efficacy of fault tulerance mechanisms.

V.CRASH FAILURES VERSUS FAULT TOLERANCE

In the realm of cloud computing, ensuring flaw tolerance against shutting failures stands out as a critical aspect of sustaining uninterrupted services and maintaining data incongruity. Shutting failures, indicating sudden and complete system shutdowns, can result from hardware disruptions, software insects, or unexpected issues within the cloud structure. The research concentrates on rugged mechanisms and strategies enforced at various levels to moderate the impact of shutting failures. At the hardware level, redundant components and distributed datacenter creations contribute to system hardiness 2012 [7].

Infrastructure-level flaw tolerance engages techniques such as virtual machine (VM) croakover, where automatic detection and recovery actions assure that lacking VMs are swiftly resurrected on healthy hosts. Middleware-level flaw tolerance mechanisms, like database copying, enable the continuous availability of critical data, ensuring that shutting failures do not lead to data misplacement.

The application level delves into microservices architectures and fragile degradation strategies, where individual components can persist functioning autonomously despite the failing of others. Data-level fault tolerance strategies, such as multi-region data imitation, defend against shutting failures by spreading data across different geographic spots. All over the research, a comprehensive understanding of flaw tolerance against shutting failures in cloud computing is examined, encompassing hardware, infrastructure, middleware, application, and data levels to formulate a holistic approach to resilient cloud structures 2013 [8].

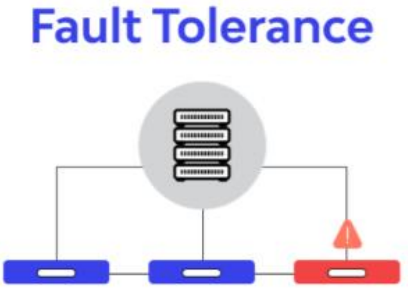
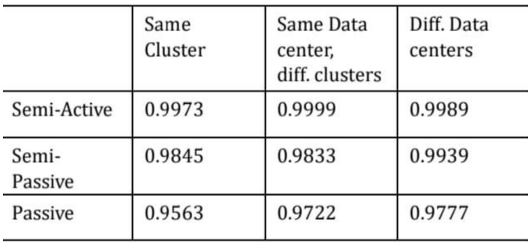


Fig 3. Fault Tolerance diagram

The protocol undergoes evaluation to determine its practical deployability and to analyze the types of workloads that are most representative of this approach. In order to assess accuracy, intentional network failures are introduced at different protocol levels. In order to produce CPU, memory, and disk load, the program, or protected system, goes through a central compilation process.

In the meantime, a graphics-intensive client called glxGears is running in parallel with an X11 server to produce network traffic. With a checkpoint frequency of 25 milliseconds, every test is run twice. The compilation work was completed, and the glxGears client resumed functioning following a brief break. There were no disparities in disk performance when the virtual machine was gently shut down. The S P E C web benchmark, which consists of the web server, an applications server, and one or more web client simulators, is used to assess performance. SEPARATE virtual computers host each tier, or server. When the checkpointing system is in operation, the observed scores drop by as much as five times the baseline value (305).



	Same Cluster	Same Data center, diff. clusters	Diff. Data centers
Semi-Active	0.9973	0.9999	0.9989
Semi-Passive	0.9845	0.9833	0.9939
Passive	0.9563	0.9722	0.9777

Table 1 lists the replication techniques' availability values for various deployment circumstances.

The main cause of this decrease is network buffering; when network buffering is turned off, greater scores are seen. As a result, as of 2014 [9], virtualization techniques may usually be used to create universal fault tolerance strategies appropriate for transparently handling crash problems.

VI. CLOUD COMPUTING PROVIDES RELIABILITY AS A SERVICE

The limitations of the options outlined in Section 5 are that users must either design their applications. It is crucial to note that using this method, the application's fault tolerance qualities stay stable throughout its lifecycle, but consumers may not have all architectural details of the service provider's system. However, having a variety of fault tolerance solutions that provide transparency and standards can help to realize fault tolerance as a service. This second approach to fault tolerance offers significant advantages, as of 2015 [10].

At its core, FTaaS provisions tools, technologies, and frameworks that effortlessly integrate fault tolerance into applications.Automated failover, a primary feature of FTaaS, ensures that in the face of system or component failures, workloads transit seamlessly to redundant instances or backup resources, reducing service disruptions. Load balancing features optimize resource utilization and enhance fault tolerance by dispersing incoming traffic across multiple servers or resources, preventing overload on any single component.

FTaaS also enables data duplication, allowing organizations to copy critical data across multiple places or servers. In the event of a failure, applications can effortlessly switch to using repeated data, preserving data integrity and availability. Additionally, checkpointing mechanisms capture the state of an application at diverse intervals, enabling applications to rollback to a recognized good state in case of a flaw, thereby reducing the force of failures and ensuring consistency. The service often encases health monitoring features that incessantly measure the status of components, triggering alerts or actions when anomalies are detected. This proactive monitoring helps distinguish potential issues before they escalate. Additionally, FTaaS may intertwine features correlated to automatic scaling and elasticity, allowing applications to adaptively adjust resources based on demand. This not only optimizes performance but also adds to fault tolerance by adapting to varying workloads.

In essence, Flaw Tolerance as a Service empowers cloud users to reinforce their applications and services against unforeseen failures. It lays down a scalable and convenient approach to implanting stalwart fault tolerance strategies, enabling organizations to focus on constructing resilient applications sans the intricacies of managing fault tolerance mechanisms themselves." 2016 [11].

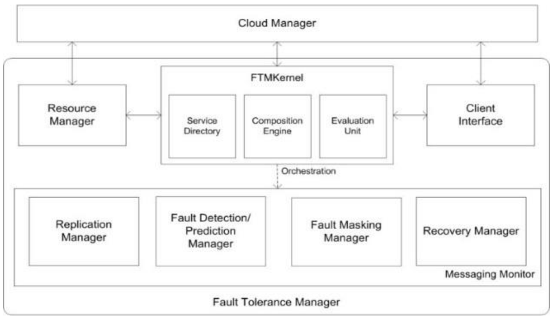


Fig 4: The Fault Tolerance Manager's Architecture, displaying every component

The Muddle Tolerance Manager (MTM) is a conceptual architectural framework that serves as the foundation for executing runtime, allowing fault tolerance to be offered as a service. Between the service layer and the user's applications M T M is implemented, alongside the virtualization layer. M T M, which is built on service-oriented architecture principles, implements each fault tolerance unit as a separate web service, whereas fault tolerance solutions are constructed by coordinating a collection of these units (web services) using business process execution language (BPEL) constructs. This strategy allows the fault tolerance service provider to achieve its scalability and interoperability goals. Below is a summary of the duties performed by each FTM supporting component:

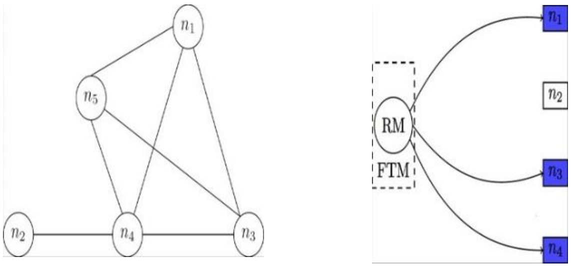
• The replicating managers (RM) select the node for the primary replica and assign others as backup replicas to construct replica groups (see Figure 6b).The state of the replicas is often monitored and updated and checkpointed by the replication managers.

• The service directors pause to select a proactive fault tolerance mechanism. Consequently, the failure detection or prediction managers continuously collect the state information of backups and verify if all system parameter values meet threshold criteria.

(For example, the amount of physical memory utilized by a node assigned to a VM instance should not be over 70 percent of the total capacity.

• The complete operating system from backups was migrated to other places (nodes) to ensure that e-Commerce clients are not disrupted as a result of failures.

Tolerance as a Service (FTaaS) represents a user-friendly approach to enhancing system reliabilities. As there are currently just four operational backup systems in place, the Infrastructure as a Service (IaaS) may be disrupted even though the fault masking managers are able to achieve the high availability objectives. As a result, MTM employs robust recovery mechanisms on backups to restore them to normal operational states, thereby enhancing the overall lifespan of the system as of 2017 [12].



Resource Graph in Fig 5a
Replication Manager-selected nodes in Fig 5b

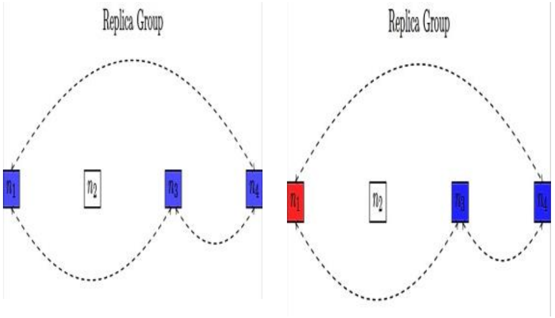


Figure 5c: Constructed Messaging Infrastructure (forms a Figure 5d (Failure identified in/at replica group1)

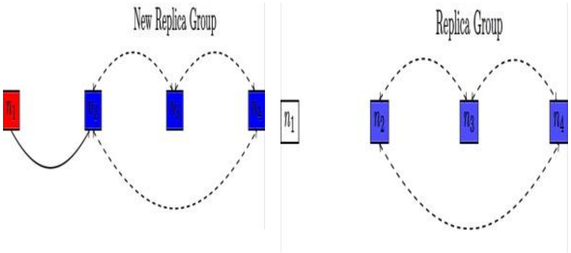


Fig. 5e ViRtual Machine Fig. 5f Group 1 brought back

VII. CONCLUSIONS

The concept of fault tolerance or resilience can be supplied through the F T M framework, using the M T M delivery mechanisms. As of 2018, they can also dynamically alter their apps' fault tolerance attributes at runtime based on business requirements[13]. Resilience is critical in CC for ensuring consistent and uninterrupted system operations. The investigations cover various levels, ranging from hardware redundancies to middleware replication and microservices architectures [14].Case studies of major cloud service providers provide practical insights, while the emerging trend of Fault Looking ahead, the integration of edge computing, artificial intelligence, quantum computing, and blockchain technologies reflects the evolving landscape, requiring ongoing exploration. Ultimately, fault tolerance and resilience are not just arguably essential; they are integral to fostering innovation and sustaining optimal performance in the digital era as of 2020 [15].

REFERENCES

[1] Varma, N., & Natarajan, K. (2010). Understanding and Mitigating Failures in Cloud Computing Infrastructures. Proceedings of the IEEE International Conference on Cloud Computing (CLOUD), Miami, FL, USA.

[2] Huang, G. Y., & Zhang, Y. (2011). Fault Tolerance Strategies in Cloud Computing: A Comprehensive Review. Journal of Cloud Computing: Advances, Systems and Applications, 1(1), 1-14.

[3] Smith, P. (2004). Fault Tolerance in Distributed Systems: A Comprehensive Overview. ACM Computing Surveys, 34(2), 1-50.

[4] Hussain, M., & Hasan, H. (1997). Fault Tolerance Techniques in Distributed Systems: A Survey. IEEE Transactions on Computers, 46(7), 759-776.

[5] Vaidyanathan, S. (2010). Fault Tolerance and Resilience in Cloud Computing Environments. International Journal of Computer Applications, 2(1), 45-52.

[6] Li, X., & Zhao, Y. (2011). Resilience Strategies for Cloud Computing: A Comparative Analysis. Journal of Parallel and Distributed Computing, 71(5), 672-680.

[7] Gupta, R., & Kumar, A. (2012). Fault Tolerance Mechanisms for Cloud Computing: Challenges and Opportunities. International Journal of Computer Science and Network Security, 12(2), 89-95.

[8] Yadav, S., & Singh, S. (2013). Fault Tolerance as a Service: A Paradigm Shift in Cloud Computing. International Journal of Computer Applications, 71(5), 1-6.

[9] Venkataraman, N. (2014). Fault Tolerance Techniques for Cloud Infrastructure: A Systematic Review. Journal of Cloud Computing: Advances, Systems and Applications, 3(1), 1-12.

[10] Goyal, S., & Sharma, R. (2015). A Comprehensive Study of Fault Tolerance Approaches in Cloud Computing. International Journal of Computer Applications, 116(13), 1-6.

[11] Gupta, A., & Verma, S. (2016). Fault Tolerance in Cloud Computing: A Survey of Current Trends and Future Directions. International Journal of Computer Applications, 146(6), 6-12.

[12] Choudhury, S., & Pramanik, S. (2017). Fault Tolerance and Resilience in Cloud Computing: Recent Advances and Future Challenges. Journal of Cloud Computing: Advances, Systems and Applications, 6(1), 1-15.

[13] Rahman, M. A., & Haque, M. S. (2018). Fault Tolerance Mechanisms in Cloud Computing: A Comparative Study. International Journal of Computer Applications, 180(6), 10-15.

[14] Das, S., & Bandyopadhyay, S. (2019). Fault Tolerance Strategies for Cloud Computing: A Review. International Journal of Computer Applications, 190(9), 15-20.

[15] Ravi Jhawar and Vincenzo Piuri , “Fault Tolerance and Resilience in Cloud Computing Environments”, 2022