# Accepted Manuscript


A Survey of Fault Tolerance in Cloud Computing

Priti Kumari, Parmeet Kaur

PII: S1319-1578(18)30643-8
DOI: https://doi.org/10.1016/j.jksuci.2018.09.021
Reference: JKSUCI 517

To appear in: *Journal of King Saud University - Computer and Information Sciences*

Received Date: 22 June 2018
Revised Date: 24 August 2018
Accepted Date: 25 September 2018

Please cite this article as: Kumari, P., Kaur, P., A Survey of Fault Tolerance in Cloud Computing, *Journal of King Saud University - Computer and Information Sciences* (2018), doi: https://doi.org/10.1016/j.jksuci.2018.09.021

# A Survey of Fault Tolerance in Cloud Computing

***Abstract***
Cloud computing has brought about a transformation in the delivery model of information technology from a product to a service. It has enabled the availability of various software, platforms and infrastructural resources as scalable services on demand over the internet. However, the performance of cloud computing services is hampered due to their inherent vulnerability to failures owing to the scale at which they operate. It is possible to utilize cloud computing services to their maximum potential only if the performance related issues of reliability, availability, and throughput are handled effectively by cloud service providers. Therefore, fault tolerance becomes a critical requirement for achieving high performance in cloud computing. This paper presents a comprehensive overview of fault tolerance-related issues in cloud computing; emphasizing upon the significant concepts, architectural details, and the state-of-art techniques and methods. The objective is to provide insights into the existing fault tolerance approaches as well as challenges yet required to be overcome. The survey enumerates a few promising techniques that may be used for efficient solutions and also, identifies important research directions in this area.
**Keyword-** Cloud Computing; Fault Tolerance; Data Center; Network Topology

## 1. Introduction

Cloud computing refers to accessing, configuring and manipulating the resources (such as software and hardware) at a remote location [1]. R. Buyya et al. [2] defined the Cloud computing in terms of distributed computing "A Cloud is a type of parallel and distributed system containing a set of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers".

According to the U.S. National Institute of Standards and Technology (NIST) definition: "Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (for example servers, networks, storage, services, and applications) that can be quickly provisioned and released with least management effort or service provider interaction" [3].

Cloud computing offers various resources in the form of services to the end users on-demand basis. It enables businesses and users to use applications without installing them on physical machines and allows access to required resources over the Internet. It provides features as high performance, pay-as-you-go, connectivity, interactivity, reliability, ease of programmability, efficiency, scalability, management of large amount of data and elasticity to transform IT from a product to a service [4] [5], as depicted in Fig. 1.


Fig. 1. Cloud Computing

The cloud computing, as a fast advancing technology, is increasingly being used to host many business or enterprise applications. However, the extensive use of the cloud-based

services for hosting business or enterprise applications leads to service reliability and availability issues for both service providers and users [6] [7]. These issues are intrinsic to cloud computing because of its highly distributed nature, heterogeneity of resources and the massive scale of operation. Consequently, several types of faults may occur in the cloud environment leading to failures and performance degradation. The major types of faults [8] [9] [10] are listed as follows:

- *Network fault*: Since cloud computing resources are accessed over a network (Internet), a predominant cause of failures in cloud computing are the network faults. These faults may occur due to partitions in the network, packet loss or corruption, congestion, failure of the destination node or link etc.
- *Physical faults:* These are faults that mainly occur in hardware resources, such as faults in CPUs, in memory, in storage, failure of power etc.
- *Process faults:* faults may occur in processes because of resource shortage, bugs in software, incompetent processing capabilities etc.
- *Service expiry fault:* If a resource's service time expires while an application that leased it is using it, it leads to service failures.

Failures lead to system breakdown or shut down of a system. However, distributed computing and thus, cloud computing is characterized by the notion of *partial failures.* A fault may occur in any constituent node, process or network component. This leads to a partial failure and consequently, performance degradation instead of a complete breakdown. Though this results in robust and dependable systems, faults should be handled effectively by proper fault tolerance mechanisms for high-performance computing. Fault tolerance enables the system to serve the request even some of the components are not working properly [6] [11].

Fault tolerance (FT) is the capability of a system that keeps on performing its anticipated function regardless of faults. In other words, FT is related to reliability, successful operation, and absence of breakdowns. A FT based system should be capable to examine faults in particular software or hardware components, failures of power or other varieties of unexpected adversities and still fulfil its specification [12].

The survey makes the following contributions:

- It is a comprehensive study of fault tolerance in cloud computing systems. It discusses the taxonomy of faults, errors, and failures along with their likely causes in a cloud environment
- Along with the existing approaches for ensuring fault tolerance in cloud systems, the same has also been described for varied distributed systems including mobile computing systems. Thus, an all-rounded perspective of the problem and its challenges are presented.
- The dependency of fault tolerance approaches in cloud computing systems upon underlying network topologies of data centres is explored. The survey discusses the most common network topologies in data centers for the cloud systems and how fault tolerance approaches to leverage the same in their implementation.
- The survey lists a number of miscellaneous cloud-based problems with which fault tolerance approaches have been integrated. We specifically emphasize on the integration of fault tolerance with cloud security.
- A number of promising techniques, such as deep learning and blockchains, that may be used effectively in this domain are discussed.
- Based on the understanding of existing challenges and solutions, a few research directions have been enumerated.

The rest of the survey paper is classified as follows: An overview of cloud computing environment, deployment models, and service stack is presented in section 2. In section 3, the traditional fault tolerance approaches used in distributed systems is presented. Section 4 discusses the existing fault tolerance approaches in the cloud computing environment. Section 5 outlines the future directions for research. Finally, section 6 presents the concluding remarks.

## 2. Background of Cloud Computing

Cloud computing has evolved from efforts in various research areas of computing such as distributed computing, grid computing, virtualization techniques and SOA (Service-Oriented Architecture). Therefore, it has imbibed their features, advances, and limitations as well.

This section describes cloud computing in five dimensions: (a) Basic concepts (b) Cloud components (c) Cloud infrastructure (d) Cloud deployment model (e) Cloud service stack, as depicted in Fig. 2.

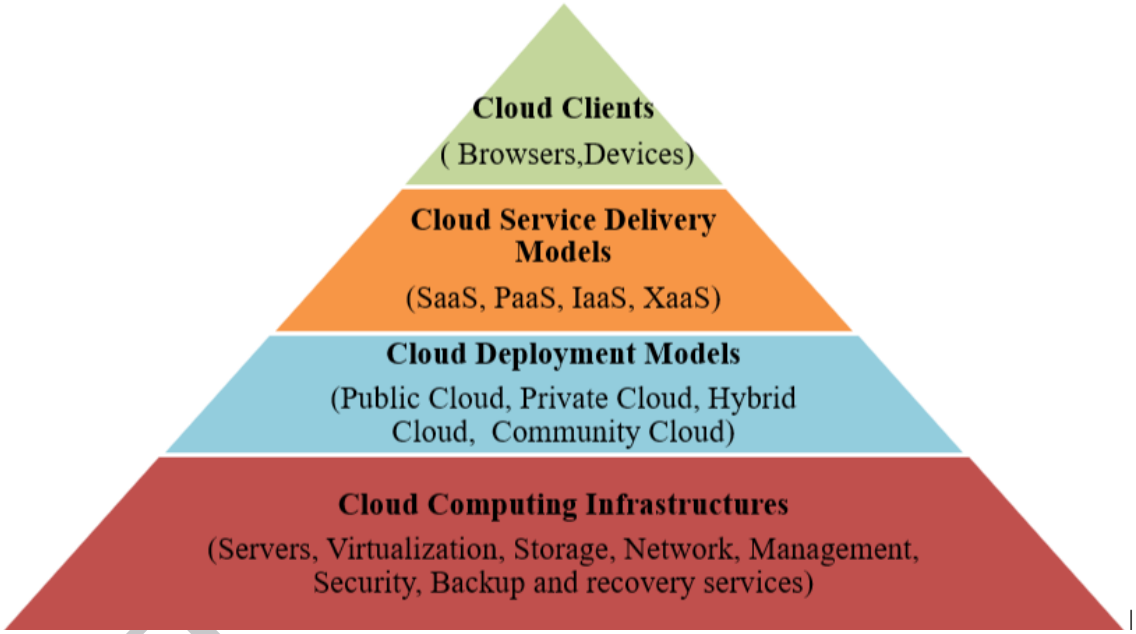


Fig. 2. Cloud Computing Architecture

### *1.1. Cloud Computing Infrastructure*

Cloud computing infrastructure includes the computers, storage devices, network facilities and other allied components necessary for offering cloud computing resources and services to users. These hardware components are mostly located within enterprise data centers. These encompass multicore servers, solid-state drives and hard disk drive offering stable storage and network devices, such as firewalls, switches, and routers; all on a large scale. Apart from these hardware components, software components that support the cloud service model, such as the virtualization software, are also termed as the cloud computing infrastructure. The virtualization software provides an abstraction of cloud resources and offers these resources to users generally using APIs (Application Program Interfaces) or other command line and/or graphical interfaces. The virtualized resources, hosted by cloud service providers (CSPs) are delivered to the users usually over the Internet (and sometimes over any other network).

Cloud computing resources are offered to users as services, in general, on a shared and multitenant-based approach. A multitenant-based approach is used by major CSPs like Amazon Web Services (AWS) and/or Google Cloud Platform. This approach is used to share resources in a cost-efficient and yet, secure manner

amongst the multiple applications and tenants (businesses, organizations, etc.) using the cloud. Virtualization software may be used to ensure isolation between the tenants. A typical cloud infrastructure comprises clients, servers, applications, and other components.

Another cloud computing component is the distributed file system (DFS), like the Google File System (GFS) and/or Hadoop Distributed File System (HDFS) which is mainly used to store data on disks in form of objects or blocks. These file systems decouple storage management from the actual physical storage and hence, ensure scalability of storage. Thus, cloud computing infrastructure consists, broadly [13], of the following:

- *Servers* – The physical machines that act as host machines for one or more e virtual machines
- *Virtualization* – Technology that abstracts physical components such as servers, storage, and networking and provides these as *logical* resources.
- *Storage* – In the form of Storage Area Networks (SAN), network attached storage (NAS), disk drives etc. Along with facilities as archiving and backup
- *Network* – To provide interconnections between physical servers and storage.
- *Management* – Various software for configuring, management and monitoring of cloud infrastructure including servers, network, and storage devices
- *Security* - Components that provide integrity, availability, and confidentiality of data and security of information, in general.
- *Backup and recovery services*

### *1.2. Cloud Deployment Models*

The cloud deployment model is based on the motive and environment in which a cloud service is expected to be used. The selection of the deployment model determines the incurred cost, power consumption by resources and other capital expenses [14]. The most commonly used deployment models in cloud environments are public cloud, private cloud, community cloud, and hybrid cloud.

- *Public Cloud:* The public cloud permits the general public to access the systems and services offered by an enterprise provider. It provides flexibility, scalability, location independence with the very low cost since multi-tenancy is generally used [1] [4] [13] [14]. Resources are dynamically provisioned on an on-demand basis from a remote third-party provider who offers resources using a multi-tenant approach.
- *Private Cloud:* The private cloud is used within a particular organization, i.e., the cloud resources and services can be accessed or used inside an organization. This model ensures high application and data security and privacy [1] [7] [13] [14].
- *Community Cloud:* This model is used by various enterprises/organizations simultaneously and helps a particular community/society that contains communal involvements (for example security necessities, mission, and compliance considerations and so on). This model may be operated, owned, and managed by one or multiple organizations inside the community or/and a third party. [3] [4] [15].
- *Hybrid Cloud:* The hybrid cloud is an alliance of both public cloud and private cloud. In this cloud deployment, critical events (e.g. those requiring secure operations) are accomplished using the private cloud services and non-critical events are implemented by using the public cloud [4] [14].

Public clouds are most suitable in scenarios where organizations wish to use collaboration services like chat, and video conferences but sufficient IT resources or

infrastructure are not available locally. In contrast, if strict security and privacy are issues of high priority, a private deployment model should be used. On the other hand, an organization that possesses a large IT infrastructure and is also expanding its capabilities, a hybrid deployment model should be the choice.

### *1.3. Cloud Service Model*

Though cloud computing has highly evolved in recent years, services are still into three major service models [1] [13] [14]. The basic service models are demonstrated in Figure 3.

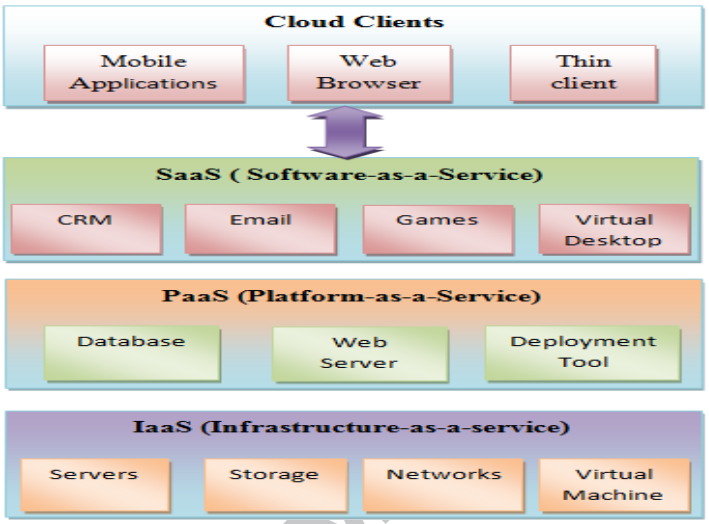


Fig. 3. Cloud Service Delivery Models

- *Software-as-a-Service (SaaS):* In this model, the software applications are presented by cloud service provider in the form of services to the consumer/end-users [1] [5] [14]. An application delivered as a service to the client removes the need to install and execute the cloud application on the user's computer and simplifies maintenance. As for example, the web conferencing services, email applications, social media platforms etc. The list of SaaS providers is Amazon AWS, Google Compute Engine, Microsoft Azure, IBM SmartCloud Enterprise, CloudStack, OpenStack, OpenNebula, CloudForge, Citrix, Qstack and so on [16].
- *Platform-as-a-Service (Paas):* This model provides a platform to develop, run, test and manage applications in the cloud [5] [14] [17]. A user can lease an environment with a software stack from a CSP and use it for custom application development. The list of PaaS providers is Acquia Cloud, Amazon AWS, App Agile, Apprenda, AppScale, Bluemix, Cloud 66, Cloudways and so on [18].
- *Infrastructure-as-a-Service (IaaS):* The IaaS model provides facility to access to some primary resources i.e. physical machines, storage, networks, servers, virtual machines on the cloud etc [5] [9] [17]. The IaaS provider offers services such as dynamic virtual machine provisioning and on-demand storage facilities. The list of SaaS providers is Salesforce, Microsoft, Amazon Web Services, Slack, Zendesk, GitHub, Oracle, Cisco and so on [19].
- *Anything-as-a-Service (XaaS):* The XaaS is another service model that may be anything or everything as a service. The cloud system is capable to maintain the huge amount of resources to fulfill the personal, granular, and specific requirements using Security-as-a-Service, Identity-as-a-Service, Communication-as-a-Service, DaaS (Database-as-a-Service) or Strategy-as-a-Service and so on [13].

## 2. Fault Tolerance Approaches in Distributed Systems

Fault tolerance is crucial for a system in order to permit it to offer the needed services even in the presence of component failures, or one or multiple faults [11] [20]. Failures in a system occur as a result of errors which are in turn due to faults (Fig 4). These are described as follows:


Fig. 4. Occurrence of failure

- *Faults:* It is the incapability of a system to perform its necessary/needed task that is caused by some abnormal state or bug present in one or multiple parts of a system [7] [8] [9] [10] [21]. Various types of faults may occur in the system, which is classified as depicted in Fig. 5.

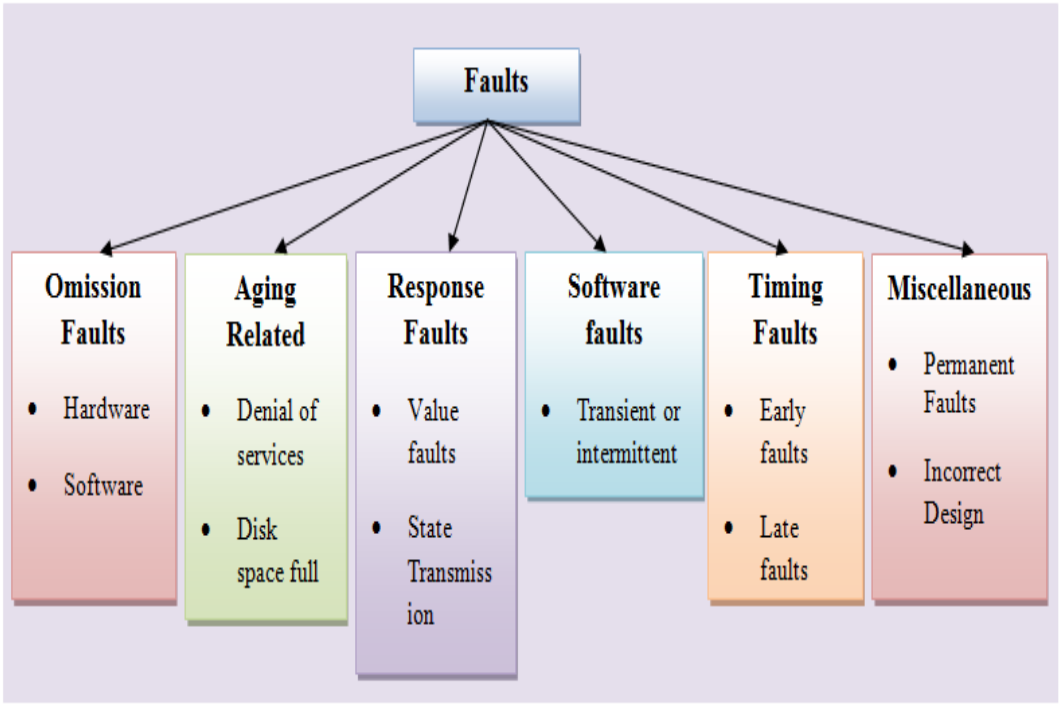


Fig. 5. Classification of Faults

- *Error:* A system component can move to an error state or an incorrect condition due to the presence of faults. A system component's performing erroneously may result in the system's partial or even complete failure [8] [21]. A distributed system can contain various types of errors, which are as shown in Figure 6.

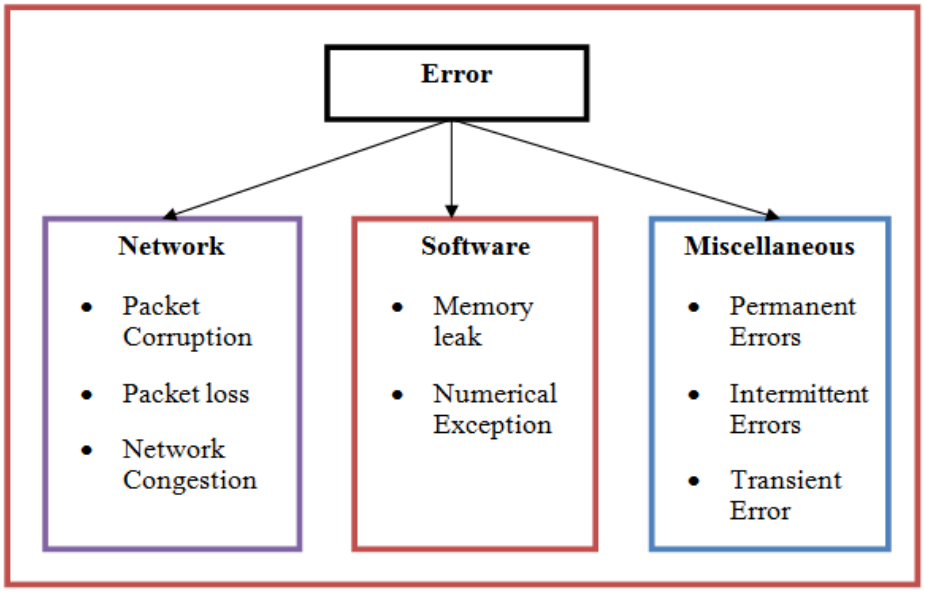


Fig. 6. Classification of Error

- *Failure*: It refers to the misbehaviour of a system that may be observed by a user (a human or some other computer system). A failure is recognized only when the system's output or outcome is incorrect [8] [21] [22]. Failures may be classified as depicted in Fig. 7.

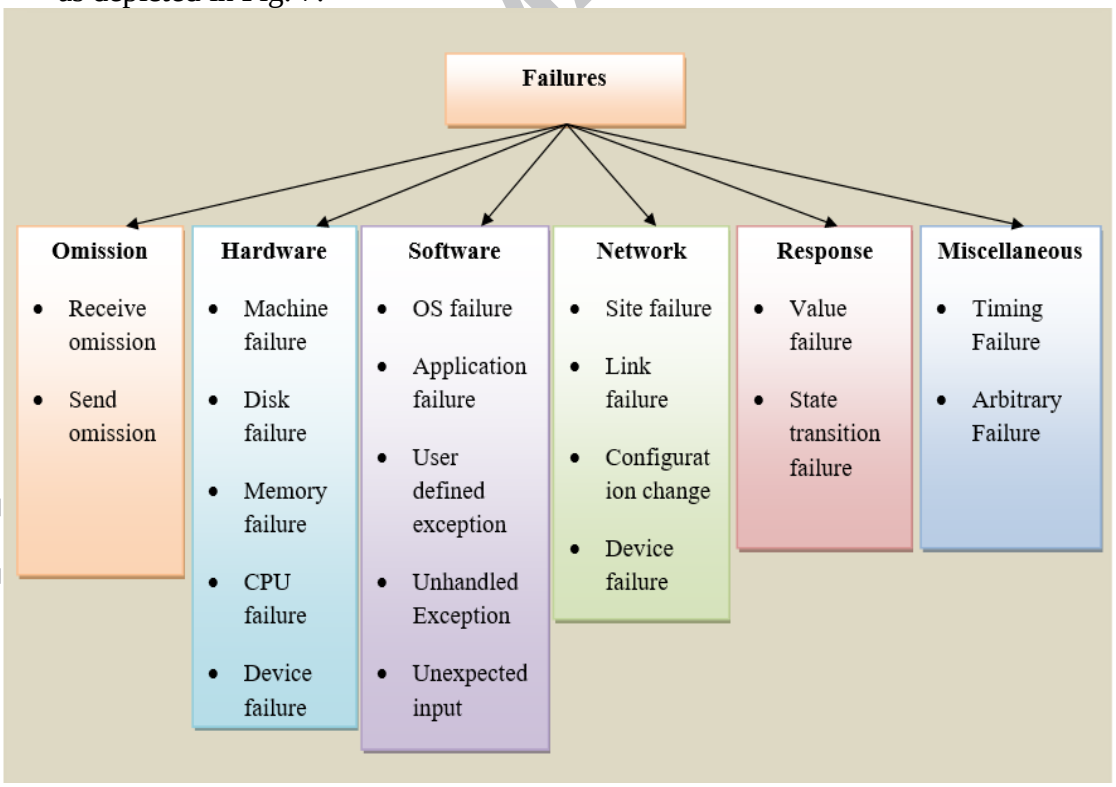


Fig. 7. Classification of Failure

Fault tolerance approaches are necessary as they aid in detecting and handling faults in the system that may occur either due to hardware (H/W) failure or software (S/W) faults. Fault tolerance is especially crucial in cloud platform as it gives assurance regarding performance, reliability, and availability of the applications. To achieve the robustness in the cloud computing, need to access as well as handle the failure

effectively [6] [7] [8] [9]. Some fault tolerance approaches, identified from literature can be categorized (Fig. 8) as follows:

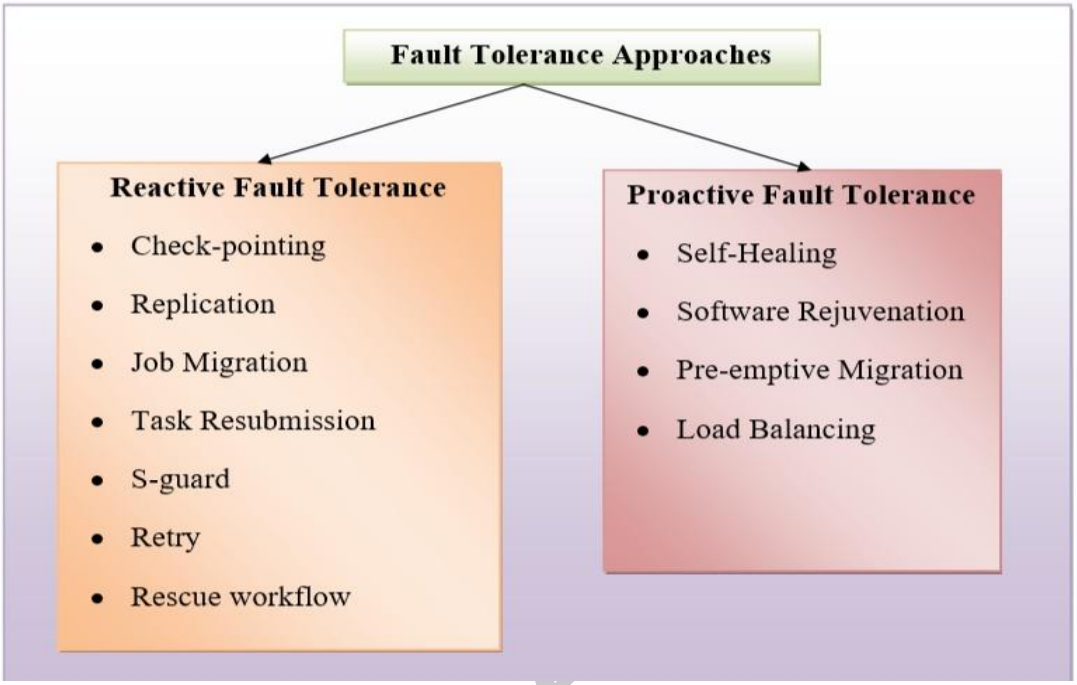


Fig. 8. Classification of fault tolerance approaches

- *Reactive fault tolerance:* This approach is mainly used to decrease the influence of failure in the cloud system after the failures/faults have actually occurred. It provides robustness or reliability to a system [9] [11]. Reactive fault tolerance approaches have been explored for cloud as well as other distributed systems. These have been listed in Table 1.

Table 1. Reactive fault tolerance mechanism with the description

| **Reactive Fault Tolerance Techniques** | **Description** |
|---|---|
| Check-pointing [23,24] | Used to save the system's state periodically. In case of a constituent task's failure, the job is restarted from the last checked pointed state rather than from the beginning. It prevents the loss of useful computation. |
| Job Migration [22] | If a job cannot complete its execution on some specific physical machine due to some reason and fails, then it is migrated to some other machine. |
| Replication [8,24] | Used to create multiple copies of tasks and store replicas at different locations. A task can continue execution in presence of malfunction or failures until all replicas are destroyed. |
| S-Guard [25] | It depends on rollback and recovery process. |
| Retry [22] | In this approach, a task is executed repeatedly until it succeeds. The same resource is used to retry the unsuccessful/failed task. |
| Task Resubmission [8,23] | In this method, the failed task is again submitted/resubmitted to the identical resource and/or to a diverse machine for execution. |
| Rescue workflow [22] | It enables a system to continue working after the failure of the task/job till it will not be able to proceed without amending the fault. |

- *Proactive fault-tolerance:* - This approach is used to predicts the faults proactively and substitute the suspected component by some running components, i.e., it avoids recovery from faults and errors [11] [20] [26] [27]. An overview of proactive FT techniques is described in Table 2.

Table 2. Proactive fault tolerance with the description

| Proactive Fault Tolerance Techniques | Description |
|---|---|
| Self-Healing [9, 23,28] | This method uses the divide and conquers technique, where a large task is decomposed into multiple chunks. This partition is mainly done to improve the system's performance. When numerous instances of the same application run on various VMs (virtual machines), then the failure of application's instances are handled automatically. It permits the computing devices or systems to itself identity, recognizes and heal dilemmas/problems occurring, without depending on the administrator. |
| Software Rejuvenation [8,22] | In this approach, the system undergoes periodic reboots and begins from a new state every time |
| Pre-emptive Migration [25,27] | In this approach, an application is observed and analysed constantly and thus, depends upon a feedback-loop control method. |
| Load Balancing [7,14,21] | This approach is used to balance the load of memory and CPU when it exceeds a maximum/certain limit. The load of exceeded CPU is transferred to some other CPU that does not exceed its maximum limit. |

- *Parameters Used for Fault tolerance in Cloud Computing:* - The fault tolerance approaches in the cloud computing are evaluated using various parameters to check the efficiency and effectiveness of the cloud systems [22] [25]. The possible parameters is listed in the Table 3.

Table 3. Parameters for FT in cloud computing and their description

| Parameters/Metrics | Description |
|---|---|
| Adaptive | All processes are automatically executed according to the conditions. |
| Performance | Used to ensure an efficiency of the system. |
| Response Time | Total time that is taken to respond/reply to a specific algorithm. |
| Throughput | It computes the number of tasks whose implementation has been completed successfully |
| Reliability | Its main motive is to provide accurate or acceptable result in a certain time period |
| Availability | It is described as probability i.e. the system is functioning properly after it is requested/intended for use |
| Usability | a user can make use of an invention/ a product to accomplish the target with efficiency, effectiveness, and satisfaction. |
| Overhead Associated | determine the total overhead involved while executing a fault-tolerance (FT) algorithm |
| Cost-effectiveness | It is a description of the system monetarily. |

*2.1. Fault Tolerance in Distributed computing environments*

Fault tolerance (FT) is an essential concern in cloud computing platform since it enables the system to provide the required services with good performance in presence of the one or more failures of the system components [6] [20] [26]. In past, fault tolerance approaches have been applied to many different distributed computing environments, apart from cloud computing. Some of them are as follows: -

- *Wired Distributed System:* - It is a collection of autonomous computers, which appears as a single consistent system to its clients/users. All computers in the distributed system contain an individual set of resources and can share some general peripheral devices, e.g. a printer. Message passing is generally used for communication in a distributed system. Designing a distributed system is a difficult task due to the existence of components that may be located at different places/sites. One of the major challenges that the system designer has to face is providing fault tolerance (FT).

  In general, FT is highly required in distributed network systems, especially in the large-scale environment. Users of a distributed system require the system to stay working continuously even in case of technical failures. If one or more of the members of the system have crashed, even then the system should be able to fulfil the client's requests. Therefore, an efficient system must be designed and implemented to handle the partial failure of its components. Failure detection (FD) and process monitoring are the most common techniques for FT in the distributed systems. Reactive fault tolerance approaches such as checkpointing, replication, retry, resubmission, etc have been used to handle failures in these systems [29] [30].
- *Mobile Computing System:* - It is a type of distributed system, where some or all the constituent nodes are mobile computers**.** This system preserves continuous network connectivity even in the existence of mobility of the hosts due to which their site/location within the network may vary with time. Each node in the system works independently, with infrequent asynchronous message communication. The

fixed nodes in the mobile system may be interconnected using a static network. Moreover, a fixed node (commonly the mobile base station) is used to establish the communication between the connected nodes, i.e., the mobile node and the other nodes inside the system. Nodes in the mobile system communicate with each other using messages [31] [32].

Some restrictions of mobile systems are limited bandwidth, mobile hosts with limited disk space, user mobility, narrow battery life, etc. To overcome the limitation of the mobile computing system, some fault tolerance approaches are used. The most commonly used FT approaches in the mobile system is check-pointing since limited resources prevent the use of redundancy-based schemes as replication etc. The FT technique needs the processes that are check-pointed at regular intervals to move the error-free state to some storage (fixed/constant). If any failure arises in the process, then that failure may be recovered by finding the newest saved/maintained state (called rollback recovery). [33]

Checkpointing schemes can be categorized as coordinated, communication-induced, and uncoordinated checkpointing. In a coordinated method, the processes adjust their checkpointing actions through transferring the checkpoint-based coordination messages. The coordinated check-pointing policies include huge message overhead, therefore not appropriate for the mobile systems and have very less bandwidth wireless communication channels/stations in the network. Furthermore, at the time of checkpointing coordination, the process execution may also require to be suspended, which may result in degradation of the performance.

The uncoordinated checkpointing methods permit processes to receive checkpoints at regular intervals without synchronization with others [34] [35] but this method may suffer from the domino effect. A communication-induced check-pointing approach has been used to handle the domino effect [31].

- *Mobile-Grid Computing:* - Grids are very large-scale systems, distributed in nature. These spread the amount of work to be done among the constituent systems. Grid computing facilitates the sharing of large-scale resources between loosely coordinated, distributed systems to solve the computational needs of large-sized tasks. Therefore, grid computing provides users with huge computational, bandwidth resources and storage. It is also possible to use grid computing in conjunction with mobile computing to get better performance. This approach is also important to handle essential restrictions of mobile devices effectively [2] [36]. However, incorporation of mobile and grid devices for use of computing resources is challenging because of unreliable connections, random node mobility, battery dependence, small-bandwidth for communication, restricted power for processing and fixed storage. An effective execution of the distributed applications in mobile grid computing (MoG) is desirable if the faults/failure of the mobile devices are handled properly [37]. Therefore, FT policies are required to handle the different types of faults in the MoGs. The most frequently used FT technique in MoGs are checkpointing and the rollback recovery. These FT methods have been used extensively in conventional wired and cellular portable distributed systems. P. Jaggi et al. [37] presented an adaptive approach for MoG based on checkpointing to recover the failure of mobile nodes.

  P.J.D. Darby et al. [38] presented ReD (Reliability Driven) middleware approach, which enabled the mobile grid scheduler to build informed conclusions/decisions, selectively submitting work portions to hosts having improved check-pointing arrangements that guarantee successful completion.

- *MANET (Mobile ad hoc networks(N/W)):* - It is a wireless ad hoc network which is self-configuring, and does not depend on infrastructure, i.e., it is an infrastructure-less network of mobile devices connected in a wireless manner. All devices in the MANET are autonomous and can change their path and direction dynamically, and therefore alter links to the other devices frequently. MANETs are widely used for increasing the computing abilities to existing mobile systems and MoG (mobile grids computing). But, MANETs are vulnerable to several transient/temporary and permanent failures. To handle the failure, an FT technique is required to be used effectively. The checkpointing, as well as rollback recovery, is a widely used policy to handle faults in static and/or cellular mobile systems. However, the use of FT approaches with MANETs are less examined. The preceding recovery-based approaches are not applied to the MANETs directly because of some challenges such as deficiency of the static infrastructure, regular movement of a node, limited bandwidth as well as the restricted amount of stable storage. To handle faults in MANETs, rollback recovery protocol based on checkpointing has been proposed, which determines the frequency of checkpoint of a mobile device/node depending on the mobility/portability and therefore avoids unnecessary checkpoints [39][40].

## 3. Fault tolerance approaches in the cloud computing

### *3.1. System model*

Cloud computing provides various services and scalable computing resources through the internet [7]. On the provider's side, a DC (data center) provides facility to keep computer systems as well as their associated components, like networking, storage, uninterruptible power supply etc [41] [42] [43]. To provide services to the clients, many virtual machines (VMs) run on the physical machines in cloud DC. These DCs use different types of network topologies. Fault tolerance approaches in cloud computing systems are dependent on underlying network topologies. In this section, we discuss basic common network topologies in DCs for the cloud system as well as reactive and proactive FT approaches used in the cloud.

The network topology is the arrangement of nodes within a network [44]. In other words, topology is a basic building block of the network that connects computer systems to each other [45]. The basic network topologies are the bus, ring, star, mesh, tree, and hybrid topology [46] (Refer Table 4).

- *Bus Topology:* - This topology is used to connect all computers and network devices through a single cable. It transfers data in one direction. This topology is very cost effective, used in small networks, easier to understand and expand. However, in this topology when cables get fail, the entire network fails [47]. If the network(n/w) is heavy then the performance of this topology is degraded. The cable contains a limited amount of length. This topology slower as compared to ring topology [45] [46].
- *Ring Topology:* - In this topology, computer systems are connected to each other in a ring structure, where the last device is connected to the first one [48]. This topology is very cheap to install as well as expand. In case of heavy network traffic and the addition of some extra node, the transmission network is not affected. However, the failure of one computer system can affect the entire network. The network activities are disturbed by adding or deleting the computers. Troubleshooting is also very difficult in the ring topology [45] [46].
- *Star Topology:* - This topology is used to connect all computer systems to a single hub with the help of a cable. This hub is used as a central node and all other available nodes are linked to it. This topology provides fast performance, easier to

troubleshoot, set up and modify. However, this topology is expensive to use and if the hub gets fail, then the entire network stopped working. The installation cost is high [44] [45] [48].

- *Mesh Topology:* - In this topology, all the node or computer systems are fully linked to each other. This topology is very robust, easier to diagnose the failure. It also provides privacy and security. However, this topology is very difficult to install or configure. The cost of cabling is also high and it requires bulk wiring [44] [45] [47].
- *Tree Topology:* - This topology contains a root node and all other computers or nodes are linked to it (known as hierarchical topology). The minimum level of the hierarchy should be three. This topology is an extended version of bus and star topology. It is also easier to manage and maintain. Error detection is also done easily in this topology. However, this topology includes a costly process and is heavily cabled [44] [46] [47].
- *Hybrid Topology:* - This topology is a combination of two or more than two topologies. This topology provides features like reliability, scalability, and flexibility. However, its design is complex and involves a costly process [46] [47].

Table 4. Summary of basic network topologies

| Topologies | Features | Advantages | Disadvantages |
|---|---|---|---|
| **Bus Topology** | • It transfers data only in a single direction.<br>• All computer devices are linked to a one cable | • This topology is cost effective.<br>• Require less cable as compared to other topologies.<br>• Suitable for small networks<br>• Easier to understand<br>• Easier to expand by merging two cables together | • When cables get fail then the entire network fails.<br>• In case of heavy network traffic or by adding more nodes, the network's performance is degraded<br>• Cable length is limited<br>• Slower than another topology |
| **Ring Topology** | • This topology makes use of repeaters with a huge number of the nodes.<br>• Data is transmitted bit by bit i.e. in a sequential manner | • The network transmission is not get affected by adding multiple nodes or due to high traffic<br>• Very cheaper to expand and install | • Troubleshooting is very critical.<br>• Addition or deletion of computer nodes may disturb the activity of the network.<br>• If one computer node fails then disturbs the entire network. |
| **Star Topology** | • All nodes are connected to the central hub.<br>• Hub is used as a repeater and help in the data flow.<br>• Used with optical fiber, twisted pair or coaxial cable. | • The performance will be fast in case of few computer nodes and very less network traffic.<br>• Easier to upgrade hub<br>• Easier to troubleshoot<br>• Easier to modify and setup. | • Installation is costly<br>• Very expensive to use<br>• If the central hub fails then the entire network stops working.<br>• Performance depends on the hub |
| **Mesh Topology** | • Fully connected<br>• Robust<br>• Not flexible | • Each connection cable can transmit its own data load<br>• It is very robust<br>• The fault diagnosis can be done easily.<br>• Provides privacy and security | • Difficult to install and configure<br>• Cost of cabling is high<br>• It requires bulk wiring |
| **Tree Topology** | • Ideal if workstations /computer nodes are placed in groups.<br>• Mainly used in WAN (Wide Area Network) | • Expansion of bus topology and star topology<br>• Easier to expand the nodes<br>• Easier to manage and maintain.<br>• Easier to detect an error | • Heavily cabled<br>• Costly<br>• If multiple nodes are added then maintenance is very difficult.<br>• If the central hub gets fail, the whole network fails. |
| **Hybrid Topology** | • It is an association of two or more than two topologies. | • Very reliable<br>• Effective<br>• Scalable<br>• Flexible | • Complex in design<br>• Costly |

The most commonly used network topologies in data centers of cloud computing environment are as follows:

- *Fat Tree topology:* - This topology is the most widely used topology for the high-performance computing (HPC) and clustering of cloud data centers (DC). It is a bidirectional multi-stage indirect topology, which provides fault tolerance and good performance levels. However, the hardware used by fat tree topology is very costly [41] [42] [49] [50] [51].
- *RUFT (reduced unidirectional fat tree):* - The RUFT topology is unidirectional MIN (multistage interconnection network), which provides good performance similar to the fat tree with very less hardware cost. This topology has not been used by any fault tolerance (FT) approach [52] [53] [54].
- *RUFT-PL (reduced unidirectional fat tree with parallel link):* - It is based on creating duplicate copies of enterprise and the injection links. It also distributes the traffic of the network in a balanced mode to decrease the HoL (Head of the line) blocking impact between dual links. The number of switches used by RUFT-PL is similar to the number of switches in the RUFT and Fat tree topologies. The switches of RUFT-PL can also twice the amount of unidirectional ports of the RUFT switches [52] [53].
- *FT-RUFT-212 (Fault Tolerance RUFT 212):* - This topology facilitates the fault-tolerance (FT) feature by creating the duplicate copy of the links, which connect to or from the ending nodes i.e. connect the nodes in a planned way that may entail a little increment in hardware(h/w) price. It uses the similar quantity of links as in RUFT topology as well as similar connection blueprint between switches [52] [53].
- *FT-RUFT-222 (Fault Tolerance RUFT 222):* - It combines the performance features of RUFT-PL as well as the fault tolerance (FT) feature of FT-RUFT-212. It contains an FT variant of RUFT topology. It makes use of two links to create the connection between the network and the processing nodes, two (2) links used for interconnecting the switch, and two (2) links to create the connection between the last phase switches and processing nodes [52] [53].
- *Z-Fat Tree topology:* -It is an extension of the fat tree known as Z- fat tree (Zoned-Fat tree). The extension is mainly related to the utilization of extra ports for each switch by providing some additional degree of connectivity. The main purpose of Z-fat tree is to deal with the issues of scalability, FT as well as routing to accomplish low latency also high bandwidth. It deals with optimization issues related to the creation of optimal FT (fat tree) topology with minimum complexity [55].
- *Clos Network topology:* - It is a type of multistage network. It provides a non-blocking network, multistage switching architecture, which reduces the number of ports needed to establish a connection. This network topology contains three stages: ingress, middle, and egress stage. Each of these stages is prepared using a number of crossbar switches [41] [42] [56].
- *VL2 topology:* -This is an agile also very cost-efficient network(n/w) design. It is created using various switches, organized inside a Clos network topology. This topology employs Valiant Load Balancing (VLB) to distribute the traffic among network paths. It also makes use of address resolution to help great server pools [41] [42] [57] [58].
- *DCell topology:* -This topology makes use of servers through several ports also low-end small switches to make the recursively defined construction. In this topology, the main basic component is DCell0, contains *n* number of servers and single *n*-port switch. Every server in DCell0 is linked to a switch in the identical DCell0 [41] [58] [59] [60].

- *BCube topology:* -It provides a recursively defined construction, which is particularly designed to deliver modular data centers based on the container. The most basic element is BCube0 similar to DCell0, i.e. *n* number of servers linked to the single *n*-port switch. For constructing a BCube1, *n* number of additional switches are used, linking to one server to every BCube0 [41] [42] [59] [61].

*1.1. Proactive approaches*

Some proposed models and frameworks based on proactive fault tolerance are as follows:

- I. Jialei Liu., et al. [62] presented a PCFT (Proactive Co-ordinated fault tolerance) method that accepts a virtual machine (VM) harmonized/coordinated method to look for a deteriorating physical machine in the data center (DC) and then migrates virtual machines from the deteriorating physical machines to some optimal target PMs automatically. An algorithm has been also proposed for the initial virtual cluster allocation.

  The proposed approach includes two-steps: In the first step, the CPU temperature-based framework is proposed to anticipate a depreciate physical machine. In the second step, an optimal target physical machine is searched using a metaheuristic algorithm, Particle Swarm Optimization algorithm. The authors have also used suitable metrics to compute the proficiency or efficiency of the proposed approach and compared with other 5 (five) models: FF(First-fit), best-fit (BF), random first-fit (RFF), modified best fit decreasing (MBFD) and IVCA. The experimental result illustrated that the PCFT approach has less transmission overhead and the total execution time is shorter as compared to the other five algorithms because PCFT uses the improved PSO (Particle swarm optimization) algorithm. The PCFT has used less edge, aggregation, root switch and overall network resources than five approaches.
- PeiYun Zhang., et al. [63] presented an online FD (fault detection) approach called SVM-Grid. This approach has been described as significant for the stability of cloud. According to the author, several fault detection models are required to know more regarding the internal structure of the cloud system. The most frequently used models are traditional SVM (support vector machine), but it provides very low accuracy. To deal with this issue, an online fault detection model has been proposed that depends on the SVM-Grid. The SVM-Grid predicts emerging issues in the clouds. The model's input parameter has been enhanced by using grid method to accomplish fine-tuned prediction for the better/higher accuracy. Further, to enhance the performance of fault prediction and to minimize the time cost, a fine-tuned prediction algorithm also an updating FT algorithm for sample DBs (databases) have been developed. Simulation experiments have been done using a dataset of the Google corporation (Google2 application cluster). The proposed approach has also been compared to existing approaches namely back propagation, LVQ (Learning vector quantization), and traditional SVM. The experimental outcomes illustrated that the newly developed model has provided improved accuracy and lower time cost as compared to BP, LVQ, and traditional SVM.

*1.2. Reactive approaches*

Some proposed models and frameworks based on reactive fault tolerance are as follows:

- S. Wang., et al. [64] proposed an OPVMP (optimal redundant virtual machine placement) model. This approach has been developed to improve the reliability of server-based cloud services using a replication-based fault tolerance method. The

proposed approach consists of the three steps: - selection of the host server, optimal VM placement, and recovery strategy decision. A heuristic algorithm has been used for appropriate host server selection and also for optimal VM placement. The experiments to verify the benefits of the approach have been performed on the CloudSim simulator [65]. Results of the proposed approach have been compared to five other existing models. An experimental result demonstrated that the proposed approach has used the fewer network resources than other algorithms.

- Mohd. Amoon [66] has developed an adaptive model/framework to handle the fault-related issue in the cloud environment. The adaptive model contains both checkpointing and replication fault tolerance techniques to get a highly reliable platform to serve the client requests. The proposed framework consists of two algorithms: one for selecting VMs to serve the consumer's requests, and other for deciding/choosing FT method, i.e. replication or checkpointing approaches. The performance of the proposed framework has been evaluated on the basis of throughput, overheads, availability and monetary cost. To perform the simulation, CloudSim tool has been used. The performance of the proposed framework has been compared to the existing OCI (optimal checkpoint algorithm). As a result, an adaptive nature of the proposed framework has improved the performance of the cloud environment as compared to existing algorithms.
- A. Zhou., et al. [67] proposed an Edge switch failure aware checkpointing (EDCKP) model. This model is designed with the aim to enhance service reliability in the cloud computing system. Fat tree network topology has been considered and two algorithms have been proposed to address the edge switch failure. One algorithm is to select storage server for the checkpoint image and other is for the recovery server. The proposed model has been compared with other existing models such as NOCKP (No checkpoint-based method) and NDCKP (is a network topology aware distributed delta checkpoint-based technique). The simulation result illustrated that the EDCKP method has reduced the total execution time and consumes fewer network resources to get the better service reliability.
- Paul J. Darby et al. [38] presented a ReD (Reliability driven) approach for FT in a mobile grid (MoG) environment. This enabled a mobile host to transmit its check-pointed data to a selected neighbouring mobile host. The selected mobile host serves as stable point storage for the checkpoints from the approved neighbouring mobile host. The ReD approach has been used to increase the likelihood of recovery of checkpointed data. It thereby maximized the probability that a distributed application executing on the mobile grid (MoG) can be completed without supporting an unrecoverable failure. In other words, ReD, Reliability Driven middleware enabled the mobile grid scheduler in informed decision making by selectively submitting work segments to the hosts that contain best-checkpointing arrangements to guarantee successful completion. The ReD approach has been evaluated in the simulator and test-bed environments. The outcome of ReD has been compared with the RCA (Random Checkpoint algorithm). As a result, stability control enhancements to the ReD produces the greatest payoff at smaller wireless areas, resulted in superior average/normal reliability achievement and in that order less break messages. The outcome is practical also useful because, in ReD, all host can decide its neighbourhood density, during message conversation/exchange among neighbours indirectly.

- J. Zhao et al. [68] presented a JCSR (Joint Checkpoint Scheduling and Routing) method to provide flexible reliability optimization in the cloud environment. A peer-to-peer checkpointing method has been used that allows client consistency levels/points to be optimized on the basis of evaluation of their individual requirements and entirely resources available in the data center. A distributed algorithm has been designed to solve the joint optimization by dual decomposition. However, the given solution can improve the use of resources and presented a supplementary source of profits to the data center operators. The validation outcome demonstrated a significant enhancement of reliability over existing methods. A heuristic algorithm, JCSR has been proposed with peer-to-peer check-pointing, which is used to locate the sub-optimal resolution for the joint checkpoint scheduling problem and routing difficulty leveraging dual decomposition process. The motive of the proposed algorithm is to come across an individual user optimization problem.
- W. Chen et al. [69] presented task failure-modelling framework. In this framework, a hypothetical analysis has been conducted on the runtime performance to know the impact of transient failures (a failure that occurs for short periods) of scientific workflow executions. To improve the workflow's runtime performance, three fault tolerance clustering strategies have been proposed, these are selective re-clustering algorithm (a new-clustered job is used to keep the clustered job having only failed tasks.) , dynamic re-clustering algorithm (used to adjust the granularity dynamically or size of the cluster or a number of tasks inside a job) and vertical re-clustering algorithm (used to divide the clustered jobs into improved jobs that reduce the job granularity and then retries them). The WorkflowSim tool has been used for the simulation.
- W. Qiu et al. [70] presented a ROCloud (reliability-based design optimization) model/framework to enhance the reliability of application using the FT approach. This framework has been used to migrate the legacy applications on cloud. The proposed model included three steps. In the first part, the legacy application analysis has been done. In the second part, the important component ranking has been done and in the last part, an optimal FT method has been chosen automatically. However, two algorithms have been proposed for ranking the components of both ordinary applications (all components of the application can migrate to cloud) and hybrid application (only some parts/portion of their components can migrate to cloud). The important value of every component has been computed based on the construction of an application, the relationships between component invocations and failure rates of component and effects of failure. A tool developed in C++ language has been used for the simulation.
- C. Chen et al. [71] presented a KNF (k-out-of-n reliability) framework, which dealt with issues of energy efficiency as well as fault tolerance. The proposed framework has provided the facility of data fragmentation at the time of data storage by a node. Other nodes are able to fetch data consistently with nominal energy consumption and also permits mobile nodes to process the distributed data. However, energy consumption required for processing data is also diminished. This framework has been evaluated in MATLAB.
- G. Yao et al. [72] presented the ICFWS (Fault tolerance workflow scheduling) algorithm. This algorithm provides the benefits of both resubmission and replication-based FT. First, the ICFWS algorithm partitions the soft deadline of the available workflow into numerous sub-deadlines. The FT policies of replication or resubmission are selected for each task and then all available tasks

are scheduled for their initial implementation. Backup copies of the tasks are retained using the replication method. After that, an online reservation, adjustment method has been proposed to fine-tune the sub-deadlines of any non-executed task during task execution process.

The main features of the discussed models and frameworks for FT have been summarized in Table 5.

Table 5. Summary of the proposed model/framework based on proactive and reactive fault tolerance

| Proposed Framework/Model | Fault tolerance Approaches | Use of Fat-Tree Topology |
|---|---|---|
| OPVMP [64] | Reactive | Yes |
| PCFT [62] | Proactive | Yes |
| Adaptive Framework [66] | Reactive | No |
| EDCKP [67] | Reactive | Yes |
| ReD [38] | Reactive | Yes |
| JCSR [68] | Reactive | Yes |
| ROCloud [70] | Reactive | No |
| KNF framework [71] | Reactive | No |
| SVM-Grid [63] | Proactive | No |
| ICFWS [72] | Reactive | No |

*1.3. Other Miscellaneous Approaches used for FT*

Some miscellaneous approaches have also been used for integrating fault tolerance methods in cloud systems to enhance their performance and to make the systems robust. Some miscellaneous approaches are discussed as follows:

- *Machine learning based Approaches:* -Machine learning is an application of artificial intelligence, which enables systems to learn automatically and use experience for improvement without any explicit programming. Its main motive is to develop computer programs, which can access the data, and they then use that data to learn [73] [74]. These machine learning techniques have also been used to develop fault tolerance methods to enhance service reliability. In particular, machine learning is employed to develop proactive fault tolerance methods where failure prediction needs to be done before it occurs in the system based on previous data of the system for the same [75] [76] [77]. Some machine learning based algorithms are described in Table 6.

Table 6. Summary of Machine Learning algorithms

| Algorithms | Descriptions |
|---|---|
| Neural Network (NN) [90] | NN is a supervised based learning approach. With an NN, a computer system is modelled to work like a human brain or the nervous system. NN works by generating connections among processing elements. The processing elements of NN are non-linear and can be interconnected using the adjustable weights. The outcome is determined by an association and weights of connections. |
| KNN (K-Nearest Neighbour) [91] | KNN is a form of supervised learning. This algorithm is used to store the available cases and categorizes new cases based on similarity measure called distance functions. It is used to solve both classification predictive and regression predictive problems. In this approach, output interpretation consumes low calculation time and predictive power. |
| SVM (Support vector machines) [90] [92] | SVMs are a kind of supervised learning. The motive of the SVM algorithm is to find a hyperplane in N-dimensional (N stand for the number of features) space that clearly classifies the data points. This algorithm is commonly used in image classification, text and hypertext classification, handwritten text/character recognition and so on. |
| Reinforcement Learning (RL) [93] | RL is a type of ML algorithm that permits machines and software representatives to automatically determine the superlative behavior in a specific situation to maximize the performance. It is a goal-oriented learning based on the interaction with the environment. It consists of two components i.e. agent and an environment. The environment states to the object where the agent is acting on. |

- *Meta-Heuristics:* - A meta-heuristic is an advanced level of heuristic which is designed to create, find or choose a heuristic. Mainly used to direct the search procedure, the objective is to effectively examine the search space to locate the near-optimal solutions. Approaches that comprise algorithms of meta-heuristic may range from easy local search processes to the complex learning procedures [78]. Meta-heuristic algorithms also approximate the outcome and are, in general, non-deterministic. The meta-heuristics approaches are usually not considered as being problem specific and therefore have been used for the efficient solution of multiple optimization problems [79]. The most commonly used meta-heuristics algorithms are ACO algorithm (ant colony optimization), PSO algorithm (Particle swarm optimization, SCO algorithm (Social cognitive optimization) etc [80] [81]. The meta-heuristic algorithms can be applied in the many distinct areas such that optimization of function, fuzzy system control, scheduling application, cloud computing, image processing, clustering, data mining, to train artificial neural network and many more. These algorithms have also been applied to improve the service reliability, i.e. the meta-heuristic algorithms have been integrated with the fault tolerance approaches to make a reliable system or enhance its performance [62] [77] [82]. Some meta-heuristic based algorithms are described in Table 7.

Table 7. Summary of Metaheuristics algorithms

| Algorithms | Description |
|---|---|
| Ant Colony Optimization (ACO) [94] | It is an optimization system proposed in the early 90s by Marco Dorigo []. ACO is a heuristic based, multi-agent optimization method inspired by the biological systems for solving complex combinatorial optimization problems. |
| Particle Swarm Optimization (PSO) [95] | This algorithm is a self-adaptive and robust global search-based optimization approach developed by Rush Eberhart and Jim Kennedy in 1995. This algorithm is developed for the population-based stochastic optimization and simulates the common behaviour of fish schooling or bird flocking. PSO is easy to implement as only a few parameters need to be adjusted to obtain a near-optimal result. |
| Artificial Bee Colony (ABC) [96] [97] | ABC algorithm is inspired by the intelligent behaviour of the honey bees. This algorithm is a swarm based meta-heuristic method. It provides a search procedure based on population and has been used to solve many distinct types of problems. The main motive of the bee is to determine source locations of food having high nectar amount. This algorithm comprises three forms of bees: employed (search food throughout the food source inside the memory then share the information about the food source to onlooker bees), onlooker (pick good food sources i.e. founded by employed bees) and scout bees (interpreted from very few employed bees that unrestraint their food sources and look for new ones). |
| Cuckoo Search Algorithm (CSA) [98] [99] | It is an optimization-based algorithm defined in the year 2009. This algorithm is stimulated by restrict brood parasitism of a few cuckoo species by arranging the eggs in the nest of some other host birds. This algorithm can also be used to resolve multi-criteria optimization issues. This algorithm can be applied in many distinct fields such that engineering optimization, stability analysis, reliability problem and many more. |
| Bee Colony Optimization (BCO) [100] [101] | This algorithm, to deal with the combinatorial optimization problems, contains two passes, i.e., forward and backward. In the forward pass phase, the partial solution is produced with individual search and collective knowledge which is subsequently employed to the backward pass phase. In the backward pass phase, the likelihood information is utilized to create decision whether to stay to explore the present solution for a subsequent forward pass or to begin searching the neighborhood with newly selected ones. |

| | |
|---|---|
| Firefly-Based Algorithm (FA) [102] [103] | It is a meta-heuristic optimization procedure inspired by fireflies in nature. This algorithm is used to solve extremely non-linear as well as multi-modal optimization problems efficiently. The speed of convergence of the algorithm is high and this algorithm has been integrated with other optimization approaches to build hybrid tools. |
| Genetic algorithm (GA) [104] [105] [106] | Genetic algorithms are a metaheuristic inspired by the procedure of natural selection. This algorithm involves five main components: initial population (process starts with the group of individuals called population), fitness function (determine capability of an individual to participate/compete with some other individual i.e. how appropriate an individual is), selection (fittest individuals are selected and then their genes are passed to next generation), crossover (for every couple of parents to be matched, a crossover point is selected randomly from the genes i.e. it represents the mating among individuals), mutation (some of their genes having low random probability subjected to mutation i.e. some of the bits with low probability inside bit string is flipped). |
| Differential evolution (DE) [107] [108] | It is a stochastic as well as population-based optimization method developed by Storn and Price. DE algorithm is a type of evolutionary programming used to resolve optimization problems on continuous domains where every variable is denoted by the set of real numbers. This algorithm has a simple structure and provides robustness. |

- *Clustering:* -In order to implement the parallel processing application in the enterprises, clustering is used in the cloud, whereby several computers, VMs (virtual machines), servers are tightly or loosely connected to work jointly and appear as a single system to the users (known as computer cluster). Cluster computing is also used for the HPC (High-performance computing). However, the long-term trend in HPC needs increasing amounts of the node inside the parallel computing platform and thus, involves a higher probability of failure. To solve the failure probability various clustering approaches are available, which can be used with the fault tolerance approaches to make the system reliable and robust [2] [83] [84].

### *1.4. Integration of Fault tolerance in miscellaneous Cloud-based and Related Applications*

Fault tolerance methods have been applied to implement reliable cloud-based applications. Some of these are related to security, workflow scheduling, mobile data offloading, and IoT based applications, where there is an immense requirement of integration of FT approaches.

*A. Security and Fault Tolerance in Cloud Computing: -*

The development of a reliable cloud computing system should not only entail the development of techniques that tolerate benign faults in the system but should also consider the handling of malicious attacks on the system. However, till date, reliable and trustworthy systems' construction has generally been viewed from two views; either security or fault tolerance, which is orthogonal to each other. For instance, from the perspective of security, it is required that the computing facilities be secured against unauthorized access and avoid redundancy. However, fault tolerance needs

redundancy in the form of replication of nodes as well as data. Also, fault tolerance schemes usually assume non-malicious nodes in the system.

Cloud-based services are shared by the millions of consumers/ users. Due to the distributed nature of the cloud, various security issues may occur. The most common security issues that may impact a cloud user are related to data, availability, authentication and authorization, privileged user access etc [109]. A few commonly occurring attacks are listed in Table 8. Fault Tolerance approaches can be effectively used to minimize the effect of these attacks [110]. The FT methods can be applied to three levels:

- *At hardware level:* if the attack on a hardware resource causes the system failure, then its effect can be compensated by using additional hardware resources.
- *At software (s/w) level:* Fault tolerance techniques such as checkpoint restart and recovery methods can be used to progress system execution in the event of failures due to security attacks.
- *At system level:* At this level, fault tolerance measures can compensate failure in system amenities and guarantee the availability of network and other resources.

Therefore, it is imperative that viewpoints of security and fault tolerance be aligned with respect to cloud computing systems. Only then, cloud computing services will be able to gain the confidence of individual users as well as enterprises with regard to their data and computations. The integration of fault tolerance and security methods should cause a minimal overhead on system performance.

Table 8. Security Attacks in the Cloud environment

**Distributed Denial of Service** attack can affect the availability or accessibility of cloud services because of its multi-tenant architecture.

It is applied to the numerous cooperated systems with two key motives [111-114]

- To overpower the resources of the server such as CPU time or the bandwidth of the network, therefore the genuine users cannot access the resource
- To hide the identity of malicious users or attackers

| **Type of DDoS Attack** | **Description** | **Affected Cloud Computing Layer** |
|---|---|---|
| **Smurf attack** | Attacker sends a massive amount of ICMP echo requests. The sent requests are deceived i.e. IP address of source will be the IP of the victim and IP destination address will be the broadcast IP address. Finally, as an outcome, the victim will be overwhelmed with broadcasted addresses [114]. | IaaS |
| **PING of death attack** | Attacker transmits an IP packet having the size more than the maximum threshold of IP protocol i.e. 65,535 bytes. In case of managing an oversized or large size packet, the victim's machine and resources of the cloud inside the cloud environment can be affected [115]. | IaaS and PaaS |
| **IP-spoofing attack** | Transmissions of the packet among the client and cloud server can be interrupted and their headers are reformed by attackers i.e.IP source field in IP packet is affected by either a genuine and/or an unreachable/unavailable IP address. As an outcome, server will reply to a legitimate/genuine user machine, and will either affect the genuine user machine, or server may not be able to finish the transaction to an unreachable IP address, that disturbs the resources of server [114] [115]. | PaaS |
| **Buffer overflow attack** | An executable piece of code is sent by the attacker to the victim to take benefit of buffer overflow vulnerability. Finally, as an output, the attacker will control the machine of the victim. The attacker might damage either | SaaS |

| | | |
|---|---|---|
| | victim's machine or make use of the infected machine to accomplish a cloud-based DDoS attack internally [115]. | |
| **Teardrop attack** | Attackers use the “Teardrop.c" program to send illegal overlying values of the IP fragments within header of the TCP packets. As an outcome, in cloud system, the victim's machine will crash during re-assembly process [114]. | IaaS and PaaS |
| **Land attack** | Attackers send “Land.c" program to counterfeit the TCP SYN packets along with the victim's IP address inside the source and destination fields. In this case, the request will be received by the machine itself and then crash the system [114]. | IaaS and PaaS |
| **SYNFlood attack** | It occurs when the attacker transmits many packets on the server but not able to complete the procedure of 3-way handshake. In this circumstance, server hold-up to finish the execution of all those packets, that cause server not to process valid requests. Similarly, SYN flooding can also be done by transferring packets through a spoofed IP address [115]. | IaaS and PaaS |
| **Botnet-based** | Esraa Alomari et. al [116] [117] has presented a comprehensive study on Botnet-based DDoS attack and its effect on application layer, particularly on the web server. The Botnet-based DDoS attack on application layer restricts resources, revenue and produces consumer dissatisfaction between others. The main motive of this attack is:<br>a. To involve damage on victim side.<br>b. The hidden goal of this attack is personal i.e. it blocks the available computing resources or reduces the service's performance required by any destination machine. So, this attack is done for the purpose of revenge.<br>c. One more purpose is to accomplish this attack is to increase popularity in community of hacker. This type of attacks may also be performed for material gain | |

i.e. to interrupt the privacy also use available data/information for their own.

W. Timothy Strayer et al. [118] presented a method to detect botnets attack by examining flow characteristics i.e. packet timing, burst period and bandwidth for indication of botnet control action and command. The author has also created an architecture which first removes traffic i.e. unlikely to be a fragment of botnet, then categorizes the residual traffic in a set/group i.e. possibly to be portion of botnet and then associates the probable traffic patterns to find the common/shared communications which would recommend activity/action of a botnet.

**XSS (Cross-Site Scripting) attack:** In a cloud environment, XSS attack generally found on the multimedia web applications of online social network (OSN) i.e. Facebook, Linkedin, Twitter, etc. In XSS attack, the attacker inserts the untrusted JavaScript code on OSN web server. This is done by stealing the user's/handler's login identifications and other complex information such that session tokens and/or financial account data/information.XSS attack can lead to harsh consequences such that stealing the cookie, hijacking of account, misinformation, DOS (Denial-of-Service) attack etc [119].

To deal with this attack, Shashank Gupta, et al. [120] proposed an XSS-secure framework to detect and alleviates the proliferation of the XSS worms from the web application of OSN in the cloud system. The developed framework is operated in two modes: -

- Training mode: The training mode is used to create the secure sanitized JavaScript (JS) code i.e. encapsulated in the templates of the web pages.
- Detection mode: During the offline mode, the discrepancy in injected sanitizers are detected using the detection mode. This mode also perceives the injection of malignant variables as well as links to JavaScript (JS) code.

The developed framework has been implemented in Java language and its components are integrated on (VMs) of the cloud system.

B. *Workflow scheduling:* - The cloud system is widely used by researchers and scientists for the implementation of scientific workflows that perform data analysis and high throughput computing. Workflows enable easy definition of computational component, data and their dependencies in a declarative manner. It enables automatic execution of workflows, improving the performance of an application, and decreasing the amount of time that is needed to obtain scientific outcomes [85]**.** Furthermore, novel pricing models and frameworks have been established by the cloud service providers that permit users to provide the available resources and to make use of those

resources in an efficient way with major cost falls. Significant price reductions are accomplished due to poorer QoS that make them less efficient/reliable and susceptible to failures**.** To deal with failure, i.e., to make the system reliable fault tolerance approaches are considered. The most frequently used fault tolerance method in workflows is checkpointing, which can tolerate an instance failure as well as decrease the execution cost [69] [86].

C. *Mobile data offloading:* - Offloading is the process used to decrease the amount of data, which is being carried by the mobile phones or cellular bands and also to free the bandwidth for other end users. Data offloading is referred to as the offloading or migration of data or traffic to cloud servers. The mobile data offloading method is categorized as the offloading of data using tiny cell networks, using WiFi networks, opportunistic mobile networks, and using heterogeneous networks [87]. Offloading can also be done using cloud computing and virtualization techniques. It also allows mobile devices to migrate the computational element of an application to the powerful servers of the cloud. As mobile devices are portable, an unstable network connectivity of mobile can affect the offloading decision [88]. To deal with this problem, fault tolerance approaches can be applied effectively.

D. *Internet of Things (IoT) applications:* - IoT has facilitated a number of applications such as smart homes, intelligent automotive, agricultural and industrial applications and many more. IoT applications have enforced new needs on FT (fault tolerance) and energy management. In other words, in some applications, such as airplane control systems, a single or a particular fault can be very dangerous and life-threatening. FT is critical in many other IoT applications, in domains like medical strategy and automotive devices, where only one fault can lead to the serious consequences [89]. Therefore, it is very significant to detect and to correct faults in IoT applications, i.e. FT is required.

E. *Social Network (SN):* These represent a network between individuals or organizations using some medium to share interests, thoughts, and different activities. SN makes use of online media i.e. social media based on internet help to establish contacts with friends, family, customers, and clients. SN can occur for business as well as social purposes or/and both using sites like Facebook, Twitter, LinkedIn etc. SN is also an important target field for marketers seeking to involve/engage users [121].

Nowadays, the social network has also integrated with the cloud computing because cloud computing enables appropriate, pervasive and on-demand access of the network resources such that application, storage etc, which can be provisioned with very nominal management cost. In other words, the cloud provides many sustainable resources to the social network in many different areas such that online marketing, news, jobs, chatting, share a picture, audio, video etc. As the cloud environment is very prone to the failure then need to use fault tolerance approaches with the social cloud to ensure the reliable communication among the users [122].

## 2. Future Directions for Research

The cloud paradigm has become a reality and has been adopted by researchers, IT industry and other organizations. Fault tolerance approaches are required to improve the quality of service in the cloud environment. However, the existing or proposed approaches have considered, in general, only the reliability issue of the simple cloud workflows and evaluation has been done using some basic metrics, such as response time, availability, throughput and reliability. Some future directions for research in this domain are suggested as follows:

*5.1 Deep Learning*

Today, the cloud computing system has become the most attractive computing model in the field of academia and industry both. The cloud service provider which has its own data center (DC) can use virtualization approach to structure physical machines (PM) into virtual machines to offer resources, infrastructure, and services to the users in the form of utility. The main problem that faced by cloud service providers is task scheduling, prediction of virtual machine workload, resource provisioning, security issues and how to minimize energy cost. There are several different methods have been developed to deal with this issue separately, but all these methods consume more power, time, provide the result with less accuracy and quality. Each of these problems can be modelled as an optimization problem and solved more effectively using deep learning approach.

Deep learning (DL) is a subclass of the ML, which learns features from data directly. When the volume of data is enlarged, ML techniques are not appropriate in terms of the performance, so in such case, DL has been found to provide better performance in terms of accuracy [123]. DL use various layers (hidden layer) of the nonlinear processing elements for the purpose of feature extraction as well as transformation. In this approach, every consecutive layer uses the outcome of the previous layer in the form of input. DL approaches have been applied in many areas like speech and audio recognition, NLP (natural language processing), computer vision, social network filtering etc [124] [125].

It is envisaged that deep learning approaches can also be integrated with the fault tolerance methods to predict the faults and take the corresponding preventive measures.

*5.2 Blockchain*

The blockchain is one of the new emerging technologies in computer science. It is a digitized, disruptive as well as decentralized technology. It holds a chain of the blocks and each block holds information or data without any central supervision. The data or information stored inside the blocks depend on types of the blockchain. This new technology is widely used to validate transactions of digital currencies. Using this approach, the authenticity of everyone present in the block can be verified even though there is no any centralized authority [126] [127]. There are many different types of blockchains available, which are defined in Table 9. There are few concepts which are used by the blockchain are:

- *Decentralized*: no any central authority for supervising anything),
- *consensus mechanism:* using this mechanism the decentralized network derives to a consensus on some particular matter.
- *Miners*: users who make use of their computational power for mining the blocks.

Apart from digital currencies, this technology has been used to provide an effective security solution for multiple applications. Due to the dispersed behaviour of the cloud, many organizations or individuals using the cloud to store their data face security issues. The blockchain technology can be used to make the cloud system more secure and trustworthy. Besides, it can also be employed to solve other fault tolerance related problems like asynchronous communication, unpredictable network delay complexity, i.e. backup and delay and so on [127] [128].

Table 9. Types of Blockchains and its brief description [129]

| Types of Blockchain | Descriptions |
|---|---|
| Public Blockchain | Provide transaction within existing blocks publicly. There is no access restriction for examples: Bitcoin, Ethereum etc) |
| Consortium Blockchain | It is a semi-decentralized blockchain and is controlled by a group of members or organizations. For examples- Ripple, R3 & Hyperledger1.0 |
| Private Blockchain | Needs an invitation from the network administrator to join For examples- Multichain, Block-stack. |

Deduplication is an approach being increasingly used in cloud computing systems to reduce the data storage costs. Data deduplication method is employed for removing redundant copies of data in cloud repositories in order to the decrease the storage space, upload bandwidth and consistency of data between multiple copies [130]. The deduplication techniques are categorized in terms of size [131] as follows:

- *File-level deduplication:* - determines redundancies among different files and eliminates these replications to diminish capacity demands.
- *Block-level deduplication:* - determines and eliminates redundancies among data blocks. The file can be decomposed into the number of smaller blocks whose size can be either constant/fixed or variable. With help of fixed-size blocks calculations, of boundaries of blocks are simplified, and blocks with variable-size offers improved deduplication efficiency.

Data deduplication approach can be used for effective implementation of scalable cloud computing systems by ensuring the consistency of data. Since only one copy for each data/file is kept in the cloud system even if such a data/file is shared by a large number of users, it enhances the utilization of cloud storage. However, this reduces system reliability. To solve the reliability and privacy issue of the deduplication system, Jin Li et al. [139] have proposed a new distributed deduplication system. The authors have proposed four constructions to help both levels i.e. file and block-level of data deduplication. In this paper secret splitting method has been employed to support the secrecy of data. In this method, secure secret sharing systems have been used to divide data into fragments and stored at the distinct location. According to the authors, using the proposed model the reliability, integrity, and confidentiality can be achieved.

Implementation of fault tolerance in systems employing data deduplication can be challenging. Development of solutions that meet the reliability expectations while also decreasing storage costs and maintaining data consistency is a research topic that needs attention.

*5.4 Emphasis on Performance Issues*

- Fault tolerance approaches should be developed with a focus to solve the reliability issues of the complex workflows.
- Appropriate metrics should be designed to evaluate the robustness of a cloud system. For an instance, a metric is required to assess the overall benefit of implementing FT approach by considering the cost incurred and the achieved reliability.

- Routing methods and network topology designs of data centers should be leveraged for the development of efficient fault tolerance techniques.
- There is also a need to develop a framework/model using proactive fault tolerance methods, which can predict and locate the reason that causes the faults in the cloud.
- The work should also focus on ways of enhancing the reliability of real-time systems.
- Moreover, besides the fault, the performance variation of virtual machines also provides a negative consequence or impact on the implementation of any task workflow. Therefore, the researchers should plan to propose a robust algorithm for scheduling through elastic resource provisioning approach for workflows in the cloud environment

## 3. Conclusion

Computing in the cloud provides various features like scalability, elasticity, high availability and many more. The cloud-computing model has changed the IT industry as it brings several benefits to individuals, researchers, organizations, and even countries. Despite providing numerous advantages, the cloud system is still susceptible to failures. Failures are inevitable in cloud computing due to the scale of operation. Fault tolerance policies are commonly implemented to handle faults effectively in the cloud environment. Fault tolerance techniques help in preventing as well as tolerating faults in the system, which may occur either due to hardware or software failure. The main motive to employ fault tolerance techniques in cloud computing is to achieve failure recovery, high reliability and enhance availability.

This survey paper has discussed cloud computing concepts, its components, service model, and deployment models. The commonly used data center network topologies are also described since these affect the design of FT algorithms. Various fault tolerance techniques, models, and algorithms to improve the reliability issue of cloud services have been outlined. Techniques to enhance the performance of fault tolerance approaches, such as heuristics, meta-heuristics, clustering approaches etc. have been enumerated. Further, metrics and parameters to assess the effectiveness and efficiency of the proposed FT approaches have been discussed. Considering limitations of existing fault tolerance techniques and the emerging technologies in related domains, we have put forth some directions for future research initiatives.

## References

[1] S. Patidar, D. Rane, and P. Jain, “A survey paper on cloud computing,” *Proc. - 2012 2nd Int. Conf. Adv. Comput. Commun. Technol. ACCT 2012*, pp. 394–398, 2011.

[2] R. Buyya et al., “Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility,” Futur. Gener. Comput. Syst., vol. 25, no. June 2009, p. 17, 2009.

[3] P. Mell and T. Grance, “The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology,” Nist Spec. Publ., vol. 145, p. 7, 2011.

[4] L. Savu, "Cloud computing: Deployment models, delivery models, risks and research challenges," *2011 Int. Conf. Comput. Manag. CAMAN 2011*, 2011.

[5] M. U. Bokhari, Q. M. Shallal, and Y. K. Tamandani, “Cloud Computing Service Models: A Comparative Study,” *IEEE Int. Conf. Comput. Sustain. Glob. Dev. INDIACom*, pp. 16–18, 2016.

[6] M. K. Gokhroo, M. C. Govil, and E. S. Pilli, “Detecting and mitigating faults in cloud computing environment,” *3rd IEEE Int. Conf.* , 2017.

[7] M. Nazari Cheraghlou, A. Khadem-Zadeh, and M. Haghparast, “A survey of fault tolerance architecture in cloud computing,” *J. Netw. Comput. Appl.*, vol. 61, pp. 81–92, 2016.

[8] Z. Amin, H. Singh, and N. Sethi, “Review on Fault Tolerance Techniques in Cloud

Computing," *Int. J. Comput. Appl.*, vol. 116, no. 18, pp. 11–17, 2015.
[9] L. P. Saikia and Y. L. Devi, "Fault tolerance techniques and algorithms in cloud system," *Int. J. Comput. Sci. Commun. Networks*, vol. 4, no. 1, pp. 1–8, 2014.
[10] Y. M. Essa, "A Survey of Cloud Computing Fault Tolerance : Techniques and Implementation," vol. 138, no. 13, p. 8887, 2016.
[11] T. J. Charity, "Resource Reliability using Fault Tolerance in Cloud Computing," no. October, pp. 65–71, 2016.
[12] E. Dubrova, "Fault tolerant design: An introduction," … Microelectron. Inf. Technol. R. …, 2008.
[13] S. Singh, Y. S. Jeong, and J. H. Park, "A survey on cloud computing security: Issues, threats, and solutions," *J. Netw. Comput. Appl.*, vol. 75, pp. 200–222, 2016.
[14] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," *NCM 2009 - 5th Int. Jt. Conf. INC, IMS, IDC*, pp. 44–51, 2009.
[15] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," Futur. Gener. Comput. Syst., vol. 28, no. 3, pp. 583–592, 2012.
[16] https://www.datamation.com/cloud-computing/50-leading-saas-companies.html, Available online 2018.
[17] E. Abdelfattah, M. Elkawkagy, and A. El-sisi, "A Reactive Fault Tolerance Approach For Cloud Computing," pp. 190–194, 2017.
[18] https://stackify.com/top-paas-providers/, Available online 2018.
[19] https://stackify.com/top-iaas-providers/, Available online 2018.
[20] G. Vall??e *et al.*, "A framework for proactive fault tolerance," *ARES 2008 - 3rd Int. Conf. Availability, Secur. Reliab. Proc.*, pp. 659–664, 2008.
[21] G. Singh and S. Kinger, "A Survey On Fault Tolerance Techniques And Methods In Cloud Computing 1," vol. 2, no. 6, pp. 1215–1217, 2013.
[22] S. Prathiba and S. Sowvarnica, "Survey of failures and fault tolerance in cloud," *Proc. 2017 2nd Int. Conf. Comput. Commun. Technol. ICCCT 2017*, pp. 169–172, 2017.
[23] S. M. A. Ataallah, S. M. Nassar, and E. E. Hemayed, "Fault tolerance in cloud computing - Survey," *11th Int. Comput. Eng. Conf.*, no. 1, pp. 241–245, 2015.
[24] S. M. Hosseini and M. G. Arani, "Fault-Tolerance Techniques in Cloud Storage : A Survey," vol. 8, no. 4, pp. 183–190, 2015.
[25] A. Bala and I. Chana, "Fault Tolerance- Challenges , Techniques and Implementation in Cloud Computing," *Int. J. Comput. Sci.*, vol. 9, no. 1, pp. 288–293, 2012.
[26] M. A. Mukwevho and T. Celik, "Toward a Smart Cloud: A Review of Fault-tolerance Methods in Cloud Systems," *IEEE Trans. Serv. Comput.*, vol. 1374, no. c, pp. 1–18, 2018.
[27] C. Engelmann, G. R. Vallée, T. Naughton, and S. L. Scott, "Proactive fault tolerance using preemptive migration," *Proc. 17th Euromicro Int. Conf. Parallel, Distrib. Network-Based Process. PDP 2009*, pp. 252–257, 2009.
[28] J. Park, G. Yoo, and E. Lee, "Proactive Self-Healing System based on Multi-Agent Technologies," 2005.
[29] N. Xiong, Y. Yang, M. Cao, J. He, and L. Shu, "A survey on fault-tolerance in distributed network systems," *Proc. - 12th IEEE Int. Conf. Comput. Sci. Eng. CSE 2009*, vol. 2, pp. 1065–1070, 2009.
[30] https://www.slideshare.net/sumitjain2013/fault-tolerance-in-distributed-systems, Accessed 2018.
[31] T. Park, N. Woo, and H. Y. Yeom, "An efficient optimistic message logging scheme for recoverable mobile computing systems," *IEEE Trans. Mob. Comput.*, vol. 1, no. 4, pp. 265–251, 2002.
[32] T. Tantikul and D. Manivannan, "A communication-induced checkpointing and asynchronous recovery protocol for mobile computing systems," *Parallel Distrib. Comput.*

*Appl. Technol. PDCAT Proc.*, vol. 2005, pp. 70–74, 2005.
[33] P. Singh and G. Cabillic, "A Checkpointing Algorithm for Mobile Computing Environment," pp. 65–74, 2003.
[34] R. Prakash, S. Member, and I. C. Society, "Low-Cost Checkpointingl and Failure Recovery in Mobile Computing Systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 7, no. October, pp. 1035–1048, 1996.
[35] A. Agbaria and W. H. Sanders, "Distributed snapshots for mobile computing systems," *Proc. - Second IEEE Annu. Conf. Pervasive Comput. Commun. PerCom*, no. Cic, pp. 151–186, 2004.
[36] T. Altameem, "Fault Tolerance Techniques in Grid Computing Systems," *Int. J. Comput. Sci. Inf. Technol.*, vol. 4, no. 6, pp. 858–862, 2013.
[37] P. Jaggi and A. Singh, "Adaptive checkpointing for fault tolerance in an autonomous mobile computing grid," *Contemp. Comput. Informatics ...*, pp. 553–557, 2014.
[38] P. J. D. Darby and N. F. Tzeng, "Decentralized QoS-Aware checkpointing arrangement in mobile grid computing," *IEEE Trans. Mob. Comput.*, vol. 9, no. 8, pp. 1173–1186, 2010.
[39] P. K. Jaggi and A. K. Singh, "Movement-Based Checkpointing and Message Logging for Recovery in MANETs," *Wirel. Pers. Commun.*, vol. 83, no. 3, pp. 1971–1993, 2015.
[40] M. L. R. Chandra, "Fault Tolerance QoS Routing Protocol for MANET," 2016.
[41] Y. Liu, J. K. Muppala, and M. Veeraraghavan, "A survey of data center network architectures," *IeeeCommun. Surv. Tutorials*, vol. 15, no. 2, pp. 1–22, 2014.
[42] T. Wang, Z. Su, Y. Xia, and M. Hamdi, "Rethinking the data center networking: Architecture, network protocols, and resource sharing," *IEEE Access*, vol. 2, pp. 1481–1496, 2014.
[43] B. Wang, Z. Qi, R. Ma, H. Guan, and A. V. Vasilakos, "A survey on data center networking for cloud computing," *Comput. Networks*, vol. 91, pp. 528–547, 2015.
[44] K. Pandya, "Network Structure or Topology," *Netw. Struct. or Topol.*, vol. 1, no. 2, pp. 22–27, 2013.
[45] N. Bisht and S. Singh, "Analytical Study of Different Network Topologies," *Int. Res. J. Eng. Technol.*, vol. 2, no. 1, pp. 88–90, 2015.
[46] https://www.studytonight.com/computer-networks/network-topology-types, Accessed 2018.
[47] S. Santra and P. P. Acharjya, "A Study And Analysis on Computer Network Topology For Data Communication," *Int. J. Emerg. Technol. Adv. Eng.*, vol. 3, no. 1, pp. 522–525, 2013.
[48] R. Hegde, S. Kumar, and K. S. Gurumurthy, "The Impact of Network Topologies on the Performance of the In-Vehicle Network," *Int. J. Comput. Theory Eng.*, vol. 5, no. 3, pp. 405–409, 2013.
[49] C. Gómez, M. Gómez, P. López, and J. Duato, "A Dynamic and Compact Fault-Tolerant Strategy for Fat-tree," *Proc. IFIP Int'l Conf. Netw. Parallel Comput.*, no. January, 2006.
[50] S. Coll, F. J. Mora, J. Duato, and F. Petrini, "Efficient and scalable hardware-based multicast in fat-tree networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, no. 9, pp. 1285–1298, 2009.
[51] F. O. Sem-Jacobsen, T. Skeie, O. Lysne, and J. Duato, "Dynamic fault tolerance in fat trees," *IEEE Trans. Comput.*, vol. 60, no. 4, pp. 508–525, 2011.
[52] D. F. Bermúdez Garzón, C. G. Requena, M. E. Gómez, P. López, and J. Duato, "A family of fault-tolerant efficient indirect topologies," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 4, pp. 927–940, 2016.
[53] D. B. Garzón, C. Gómez, M. E. Gómez, P. López, and J. Duato, "FT-RUFT: A performance and fault-tolerant efficient indirect topology," *Proc. - 2014 22nd Euromicro Int. Conf. Parallel, Distrib. Network-Based Process. PDP 2014*, pp. 405–409, 2014.
[54] C. Gómez, F. Gilabert, M. E. Gómez, P. López, and J. Duato, "RUFT: Simplifying the

fat-tree topology," *Proc. Int. Conf. Parallel Distrib. Syst. - ICPADS*, pp. 153–160, 2008.
[55] M. Adda and A. Peratikou, "Routing and Fault Tolerance in Z-Fat Tree," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 8, pp. 2373–2386, 2017.
[56] Z. Dong and R. Rojas-Cessa, "Non-blocking memory-memory-memory Clos-network packet switch," *2011 34th IEEE Sarnoff Symp. SARNOFF 2011*, pp. 1–5, 2011.
[57] A. Greenberg *et al.*, "VL2: A Scalable and Flexible Data Center Network," *ACM SIGCOMM Conf. Data Commun.*, pp. 51–62, 2009.
[58] J. Zhang, F. R. Yu, S. Wang, T. Huang, Z. Liu, and Y. Liu, "Load Balancing in Data Center Networks: A Survey," *IEEE Commun. Surv. Tutorials*, no. c, pp. 1–1, 2018.
[59] B. Lebiednik, A. Mangal, and N. Tiwari, "A Survey and Evaluation of Data Center Network Topologies," pp. 1–12, 2016.
[60] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "DCell: A Scalable and Fault-Tolerant Network Structure for Data Centers," *Proc. ACM SIGCOMM 2008 Conf. Data Commun. - SIGCOMM '08*, pp. 75–86, 2008.
[61] C. Guo, G. Lu, D. Li, H. Wu, and X. Zhang, "BCube: a high performance, server-centric network architecture for modular data centers," *SIGCOMM '09 Proc. ACM SIGCOMM 2009 Conf. Data Commun.*, pp. 63–74, 2009.
[62] I. Jialei Liu, Shangguang Wang, Senior Member, IEEE, Ao Zhou, Sathish A.P Kumar, Fangchun Yang, Senior Member, IEEE, and Rajkumar Buyya, Fellow, "Using Proactive Fault-Tolerance Approach to Enhance Cloud Service Reliability," *IEEE Trans. Cloud Comput.*, pp. 1–1, 2016.
[63] P. Zhang, S. Member, S. Shu, and M. Zhou, "An Online Fault Detection Model and Strategies Based on SVM-Grid in Clouds," vol. 5, no. 2, pp. 445–456, 2018.
[64] S. Wang *et al.*, "Cloud Service Reliability Enhancement via Virtual Machine Placement Optimization Cloud," vol. 10, no. June, pp. 902–913, 2016.
[65] R. N. Calheiros, R. Ranjan, C. A. F. De Rose, and R. Buyya, "CloudSim: A Novel Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services," pp. 1–9, 2009.
[66] M. Amoon, "Adaptive Framework for Reliable Cloud Computing Environment," *IEEE Access*, vol. 4, pp. 9469–9430, 2016.
[67] A. Zhou, Q. Sun, and J. Li, "Enhancing reliability via checkpointing in cloud computing systems," *China Commun.*, vol. 14, no. 7, pp. 108–117, 2017.
[68] J. Zhao, Y. Xiang, T. Lan, H. H. Huang, and S. Subramaniam, "Elastic Reliability Optimization Through Peer-to-Peer Checkpointing in Cloud Computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 2, pp. 491–502, 2017.
[69] W. Chen, S. Member, R. Ferreira, E. Deelman, and T. Fahringer, "Dynamic and Fault-Tolerant Clustering for Scientific Workflows," vol. 4, no. 1, pp. 49–62, 2016.
[70] W. Qiu, Z. Zheng, X. Wang, X. Yang, and M. R. Lyu, "Reliability-Based Design Optimization for Cloud Migration," vol. 7, no. 2, pp. 223–236, 2014.
[71] C. Chen, M. Won, R. Stoleru, and G. G. Xie, "Energy-Efficient Fault-Tolerant Data Storage and Processing in Mobile Cloud," vol. 3, no. 1, pp. 28–41, 2015.
[72] G. Yao, Y. Ding, and K. Hao, "Using Imbalance Characteristic for Fault-Tolerant Workflow Scheduling in Cloud Systems," *Ieee Trans. Parallel Distrib. Syst.*, vol. 28, no. 12, pp. 3671–3683, 2017.
[73] Z. Chen, S. W. Son, W. Hendrix, A. Agrawal, W. K. Liao, and A. Choudhary, "NUMARCK: Machine Learning Algorithm for Resiliency and Checkpointing," *Int. Conf. High Perform. Comput. Networking, Storage Anal. SC*, vol. 2015–Janua, no. January, pp. 733–744, 2014.
[74] S. A. Macskassy and F. Provost, "A Brief Survey of Machine Learning Methods for Classification in Networked Data and an," pp. 172–175.

[75] F. Project and L. Leem, "Title : Fault-Tolerant Architecture for Machine Learning Applications."
[76] D. Kochhar, A. Kumar, and J. Hilda, "AN APPROACH FOR FAULT TOLERANCE IN CLOUD COMPUTING USING MACHINE LEARNING TECHNIQUE," vol. 117, no. 22, pp. 345–351, 2017.
[77] Z. Li-qun, L. Shu-chen, S. Cheng-li, and Z. Chun-yan, "Fault-tolerant Control Algorithm of Neural Network Based on Particle Swarm Optimization," no. 1, pp. 700–704, 2011.
[78] M. E. Frincu and C. Craciun, "Multi-objective meta-heuristics for scheduling applications with high availability requirements and cost constraints in multi-cloud environments," *Proc. - 2011 4th IEEE Int. Conf. Util. Cloud Comput. UCC 2011*, pp. 267–274, 2011.
[79] P. V. M. D, A. Umamakeswari, S. G. B, G. Shaileshdheep, V. Aishwarya, and E. R. S. Subramanian, "A Study of Various Meta Heuristic Algorithms For Scheduling in Cloud," vol. 115, no. 7, pp. 205–208, 2017.
[80] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: overview and conceptual comparison," *ACM Comput. Surv.*, vol. 35, no. 3, pp. 189–213, 2003.
[81] L. Bianchi, M. Dorigo, L. M. Gambardella, and W. J. Gutjahr, "A survey on metaheuristics for stochastic combinatorial optimization," *Nat. Comput.*, vol. 8, no. 2, pp. 239–287, 2009.
[82] N. Mohanapriya, "A Fault Tolerant Router with Optimized Ant Colony Algorithm," no. 978, pp. 167–170, 2016.
[83] T. Shwe and W. Aye, "A fault tolerant approach in cluster computing system," *5th Int. Conf. Electr. Eng. Comput. Telecommun. Inf. Technol. ECTI-CON 2008*, vol. 1, pp. 149–152, 2008.
[84] A. Patil, A. Shah, S. Gaikwad, A. a. Mishra, S. S. Kohli, and S. Dhage, "Fault Tolerance in Cluster Computing System," *2011 Int. Conf. P2P, Parallel, Grid, Cloud Internet Comput.*, pp. 408–412, 2011.
[85] D. Poola, K. Ramamohanarao, and R. Buyya, "Fault-tolerant workflow scheduling using spot instances on clouds," *Procedia Comput. Sci.*, vol. 29, pp. 523–533, 2014.
[86] W. Chen and E. Deelman, "WorkflowSim: A toolkit for simulating scientific workflows in distributed environments," *2012 IEEE 8th Int. Conf. E-Science, e-Science 2012*, 2012.
[87] H. Zhou, H. Wang, X. Li, and V. C. M. Leung, "A Survey on Mobile Data Offloading Technologies," *IEEE Access*, vol. 6, pp. 5101–5111, 2018.
[88] S. Deng, L. Huang, J. Taheri, and A. Y. Zomaya, "Computation Offloading for Service Workflow in Mobile Cloud Computing, Parallel and Distributed Systems," IEEE Trans. Parallel Distrib. Syst., vol. 26, no. 12, pp. 3317–3329, 2014.
[89] T. Xu and M. Potkonjak, "Energy-efficient fault tolerance approach for internet of things applications," *Proc. 35th Int. Conf. Comput. Des. - ICCAD '16*, pp. 1–8, 2016.
[90] M. Trivedi, "A Survey on Resource Provisioning Using Machine Learning in Cloud Computing," vol. 4, no. 4, pp. 546–549, 2016.
[91] "Available Online at www.ijarcs.info International Journal of Advanced Research in Computer Science ENHANCE DATA SECURITY IN CLOUD COMPUTING USING MACHINE," vol. 8, no. 0976, pp. 393–397, 2017
[92] Z. He and R. B. Lee, "Machine Learning Based DDoS Attack Detection From Source Side in Cloud," 2017.
[93] R. Shaw, E. Howley, and E. Barrett, "An Advanced Reinforcement Learning Approach for Energy-Aware Virtual Machine Consolidation in Cloud Data Centers," pp. 61–66, 2017.
[94] H. liu, D. Xu, H. Miao, "Ant Colony Optimization Based Service flow Scheduling with Various QoS Requirements in Cloud Computing", First ACIS International Symposium on Software and Network Engineering, pp. 53-58, 2011.

[95] S. Pandey, L. Wu, S.M. Guru, R Buyya, "A Particle Swarm Optimization-based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments". 24th IEEE International Conference on Advanced Information Networking and Applications, pp. 400-407, 2010.

[96] B. Kruekaew and W. Kimpan, "Virtual Machine Scheduling Management on Cloud Computing Using Artificial Bee Colony," vol. I, pp. 1–5, 2014

[97] W. G. Ã and S. Liu, "Computers & Operations Research A modified artificial bee colony algorithm," *Comput. Oper. Res.*, vol. 39, no. 3, pp. 687–697, 2012.

[98] M. Mareli and B. Twala, "An adaptive Cuckoo search algorithm for optimisation," *Appl. Comput. Informatics*, no. September, 2017

[99] https://www.slideshare.net/AnujaJoshi6/cuckoo-o-ptimization-ppt, Available online 2018.

[100] https://www.researchgate.net/publication/259383112_Honey_Bees_Inspired_Optimization_Method_The_Bees_Algorithm ,Available online 2018.

[101] B. Yuce, E. Mastrocinque, and D. T. Pham, "Honey Bees Inspired Optimization Method: The Bees Algorithm," no. November 2015, 2013.

[102] https://www.slideshare.net/supriyashilwant/firefly-algorithm-49723859, Available online 2018.

[103] X. Yang, "Firefly Algorithm : Recent Advances and Applications," pp. 1–14.

[104] S. Haider and B. Nazir, "Dynamic and Adaptive Fault Tolerant Scheduling With QoS Consideration in Computational Grid," pp. 7853–7873, 2017

[105] M. Lagwal, "Algorithm," pp. 560–565, 2017

[106] S. A. Hamad, "Genetic-Based Task Scheduling Algorithm in Cloud Computing Environment," vol. 7, no. 4, pp. 550–556, 2016

[107] A. Rani, "PROPOSED MODEL FOR MULTI- OBJECTIVE DIFFERENTIAL EVOLUTION ( MODE ) IN CLOUD COMPUTING," no. 3, pp. 4119–4124, 2016.

[108] C. Science and S. Engineering, "Differential Evolution Based Optimal Task Scheduling in Cloud Computing," vol. 6, no. 6, pp. 340–347, 2016.

[109] X. Chen, X. Huang, J. Li, J. Ma, W. Lou, and D. S. Wong, "New Algorithms for Secure Outsourcing of Large-Scale Systems of Linear Equations," vol. 10, no. 1, pp. 69–78, 2015.

[110] A. Mishra, "A Comparative study of Distributed Denial of Service Attacks , Intrusion Tolerance and mitigation Techniques," 2011.

[111] B. B. G. O. P. Badve, "Taxonomy of DoS and DDoS attacks and desirable defense mechanism in a Cloud computing environment," Neural Comput. Appl., vol. 28, no. 12, pp. 3655–3682, 2017.

[112] K. Chauhan and V. Prasad, "Distributed Denial of Service (DDoS) Attack Techniques and Prevention on Cloud Environment," vol. 4, no. September, pp. 210–215, 2015.

[113] B.B. Gupta "An Introduction to DDoS Attacks and Defense Mechanisms: An Analyst's Handbook". Lap Lambert Academic Pub., 2011.

[114] A. Overview, D. Attacks, and C. Environment, "An Overview of DDoS Attacks in Cloud Environment."

[115] M. Darwish, A. Ouda, and L. F. Capretz, "Cloud-based DDoS Attacks and Defenses," 2011.

[116] E. Alomari, "Botnet-based Distributed Denial of Service ( DDoS ) Attacks on Web Servers : Classification and Art," vol. 49, no. 7, pp. 24–32, 2012.

[117] M. Anbar, "A Survey of Botnet-Based DDoS Flooding Attacks of Application Layer :," pp. 2016–2018.

[118] Strayer W.T., Lapsely D., Walsh R., Livadas C. (2008) Botnet Detection Based on Network Behavior. In: Lee W., Wang C., Dagon D. (eds) Botnet Detection. Advances in Information Security, vol 36. Springer, Boston, MA

[119] P. Chaudhary, "Auditing Defense against XSS Worms in Online Social Network-Based Web Applications," no. 2014, pp. 2016–2018.

[120] S. Gupta and B. B. Gupta, XSS-secure as a service for the platforms of online social network-based multimedia web applications in the cloud. Multimedia Tools and Applications, 2016.

[121] K. Chard, S. Caton, O. Rana, and K. Bubendorfer, "Social Cloud : Cloud Computing in Social Networks," pp. 99–106, 2010.

[122] Z. Ali, R. Rasool, and P. Bloodsworth, "Social Networking for Sharing Cloud Resources,"
pp. 160–166, 2012.

[123] F. Qiu, B. Zhang, and J. Guo, "A Deep Learning Approach for VM Workload Prediction in the Cloud," 2016.

[124] S. California and L. Angeles, "DRL-Cloud : Deep Reinforcement Learning-Based Resource Provisioning and Task Scheduling for Cloud Service Providers," pp. 129–134, 2018.

[125] Q. Zhang, L. T. Yang, Z. Yan, and Z. Chen, "An Efficient Deep Learning Model to Predict Cloud Workload for Industry Informatics," *IEEE Trans. Ind. Informatics*, vol. 14, no. 7, pp. 3170–3178, 2018.

[126] D. K. Tosh, S. Shetty, X. Liang, C. A. Kamhoua, K. A. Kwiat, and L. Njilla, "Security Implications of Blockchain Cloud with Analysis of Block Withholding Attack," pp. 458–467, 2017.

[127] J. H. Park and J. H. Park, "SS symmetry Blockchain Security in Cloud Computing : Use Cases , Challenges , and Solutions," pp. 1–13, 2017.

[128] C. Xu, K. U. N. Wang, M. Guo, C. Xu, and T. K. Wang, "INTELLIGENCE IN THE CLOUD Intelligent Resource Management in Blockchain-Based Cloud Datacenters."

[129] https://www.guru99.com/blockchain-tutorial.html, Available online 2018.

[130] Z. Kalyani, A. Amune, M. Chas, and M. Chas, "REVIEW ON SECURE DISTRIBUTED DEDUPLICATION SYSTEMS WITH IMPROVE," vol. 5, no. 1, 2016.

[131] J. Li, X. Chen, X. Huang, S. Tang, Y. Xiang, and S. Member, "Secure Distributed Deduplication Systems with Improved Reliability," vol. 64, no. 12, pp. 3569–3579, 2015.

# A Survey of Fault Tolerance in Cloud Computing

Priti Kumari[1] , Dr. Parmeet Kaur[2]
Department of Computer Science & Information Technology
Jaypee Institute of Information Technology, Noida
priti19sep@gmail.com
parmeet.kaur@jiit.ac.in

## BIOGRAPHY

**Priti Kumari,** received the MCA degree from the Banasthali University of Rajasthan, India in 2014. Got the M.Tech degree in Computer Science & Engineering from the Banasthali University of Rajasthan, India in 2016. She is currently a Ph.D. student in Jaypee Institute of Information Technology, Noida, India, Computer Science branch. Her research interests include Cloud Computing, Fault Tolerance, and Distributed Computing.

**Dr. Parmeet Kaur,** received the B.E. degree in Computer Science & Engineering from the Punjab Engineering College of Chandigarh, India. She recieved the M.Tech degree in Computer Science from the Kurukshetra University of Haryana and Ph.D. degree in Computer Science from NIT, Kurukshetra, India. She is currently an Assistant Professor (Senior Grade) in Jaypee Institute of Information Technology, Noida, India, Computer Science branch. Her research interests include Mobile Computing, Fault Tolerance, Theory of computation, Compiler design and Distributed algorithms.