Trust And Fault Tolerance Models In Cloud Computing: A Review

Shivani Jaswal, Manisha Malhotra

Abstract: Cloud Computing has been considered as a future technology of internet. This is all because of sharing of IT resources, feature of scalability, flexibility and higher levels of automation. With this the rapid growth, Cloud Computing has bought concerns of security trust. Various trust issues of Cloud have been addressed by a combination of frameworks, standards and related technologies. Sometimes, consumers avoid a specific technology whenever it shows no ability to cope with their security demands. This type of loss can occur in computing platforms such as Cloud platforms and mobile platforms. Also, the concept of fault tolerance that helps in working of a system even when some of the functionalities are not working with full efficiency. Along with Trust, Fault tolerance is also a vital issue in Cloud computing platforms and applications. This feature enables any system to continue its operation at a reduced level, rather than completely failing in delivery output, especially when some subcomponent of the system malfunctions unexpectedly. This paper represents various trust and fault tolerance models existing in cloud environment along with its existing challenges.

Index Terms: Cloud Computing, Fault Tolerance, FTM, Proactive, Reactive, Trust, Trust model, TEMRT.

1. INTRODUCTION

Cloud computing is a distributed paradigm that believes in providing on-demand and dynamic provisioning of computing resources to its end-users, investing virtualization and various other Internet technologies [1]. In addition to this, the various opportunities provided by Cloud Computing are enthralling for the consumers which are ignoring presently high competitive service environments. Larger organizations tend to switch to cloud services. However, CSPs claims at providing robust security mechanisms but again numerous incidents of security breaches has been reported in past few years. The freedom provided by Cloud computing is too attractive for the consumers which can easily refuses present highly competitive service environments. The features of highly distributing and non-transparency of Cloud computing have further generated a considerable hindrance to its acceptance. Large number of users feels that they will lose their control over the saved data. More often, users are not able to trust their Cloud provider completely. [2]. To achieve its potential in cloud computing, there is a need to have a crystal clear memorandum of understanding between Cloud service provider and its consumers by taking various issues and parameters into consideration [3].Along with trust issue, the challenge of fault tolerance has its pivotal status and importance. Not because if we have best Clouds and its service available and not equipped with feature of fault tolerance, it becomes pointless to have desired cloud services from its providers. Therefore, fault tolerance feature means if a minute fault or a major failure interrupts a system, then it should be able to detect it, resolve it and should be capable of generating desired outputs [4].

- *Shivani Jaswal is currently pursuing her Doctorate degree in Cloud Computing from Chandigarh University. E-Mail id: uic.shivanijaswal@gmail.com*
- *Manisha Malhotra has done her Doctorate degree in the field of Cloud Computing. E-Mail id:mmanishamalhotra@gmail.com.*

This paper is arranged as: Section 2 gives an overview of delivery models, deployment models and some additional entities. Section 3 explains the trust and its related work along with its models and parameters. Section 4 illustrates the issue of Fault tolerance, its existing model with parameters in detail. Section 5 describes the compartitive analysis of Trust and Fault Tolerance models by showing various challenges. Finally, last section summarizes with the conclusion and its future scope.

2 CLOUD COMPUTING LANDSCAPE

This section of this paper explains the landscape from our perspective. In particular, it illustrates various following building blocks and taxonomy of Cloud Computing.

2.1 Service Delivery Models

Cloud computing is providing numerous variety of services to its users. There are many delivery models which are available. But the most basic ones are:

- Infrastructure as a Service: Examples are Simple Storage Service (S3)
- Platform as a Service: Google App Engine, Microsoft Azure etc.
- Software as a Service: Google Apps, Facebook etc.

2.2 Service Delivery Models

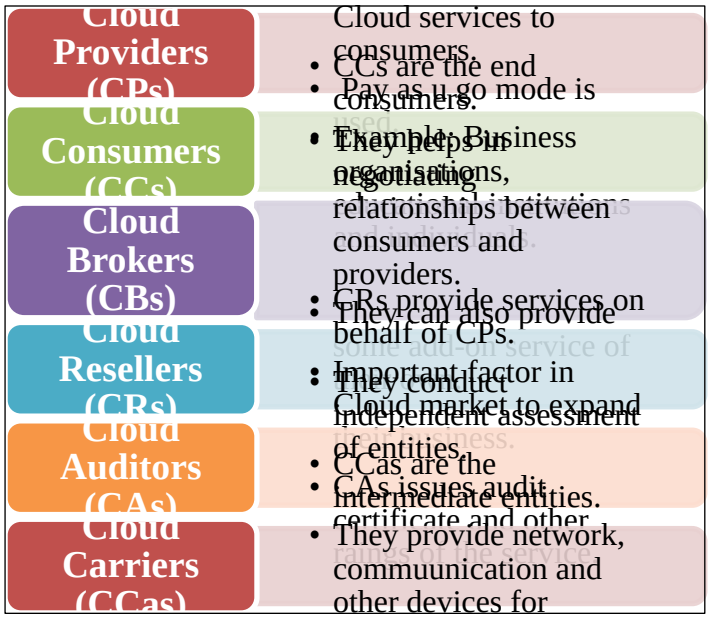
Four deployment models have been identified for cloud architecture solutions. There are four types of models where cloud services can be deployed:

- Private: It is an enterprise owned cloud that delivers its services to its limited users.
- Public: It is open for public use. Unlimited number of users can use its services.
- Community: It possesses a shared infrastructure for specific community. Hybrid: It is a combination of two or more cloud that shares the features of shared clouds.

2.3 Cloud Entities

In this, the primary entities in the market are Cloud service providers and their consumers. Additionally, there are other two emerging entities in this market i.e. brokers and resellers. Also, as per the laest refernce from NIST, Cloud Auditors and Cloud Carriers are the further two entities [2]. Following figure 2.3 visualizes the various cloud entities that are available in

market nowadays:
The Cloud landscape presented here gives a clear idea about various structure that is followed in Cloud computing. Further, this paper will elaborate various taxonomies of trust and fault



tolerance.

Figure 2.3: *Cloud Entities*

3. TRUST IN CLOUD COMPUTING

Trust is considered as one of the most vital and complex relationship between any two entities because of its some highly acknowledged properties. Comprehensively, trust implies a demonstration of trust; certainty and unwavering quality in something that is required to carry on or convey as guaranteed. Nowadays, cloud consumers are highly concerned about the capabilities of cloud providers. Moreover, the Cloud services users are not able to know the physical location of the stored of the data and various other security challenges [5]. Additionally, a market survey [6] was conducted in which not less than 3000 Cloud consumers from various geographical locations reveals that almost 84% of the Cloud consumers are having concern about their storage location and almost 88% of the consumers are worried about who has access to their data. The Cloud market is growing at a pace with entrance of new players [7]. Therefore, it is highly recommended to identify cloud service provider which can be further trusted on various parameters. Trust is a relationship which has to be maintained among two parties i.e. trustor and a trustee. The trustor (or assessor) is a type of party that helps in evaluating trustworthiness of the trustee. The trustee is a type of party which is kept under evaluation of its trustworthiness. The trust relationship between trustor and trustee is always specific in many contexts. Basically, evaluation of trust includes the answering of a question, “That while initiating an interaction session in a system, which nodes a user should interact (based on trust) and which should not be interacted?”
Assuming there are two parties A & B.

B is using a service X.

At that point A will trust on party B (as per the trust value generated by B) for utilizing services of X.
In next section, we will discuss about various trends on the basis of which trust can be build and is helpful in increasing the services to be used by Cloud consumers as provided by the Cloud providers.

3.1. Trends in Trust establishment

There are numerous quantitative approaches on the basis of which any consumer is selecting Cloud providers [2]. Some of them are Service Level Agreements (SLA), Audits, measurements, ratings by self-assessments mechanisms. We have further classified and analyzed these approaches through parameters. The figure 3.1 represents various approaches in which improvement scan be done in context of trust establishment.

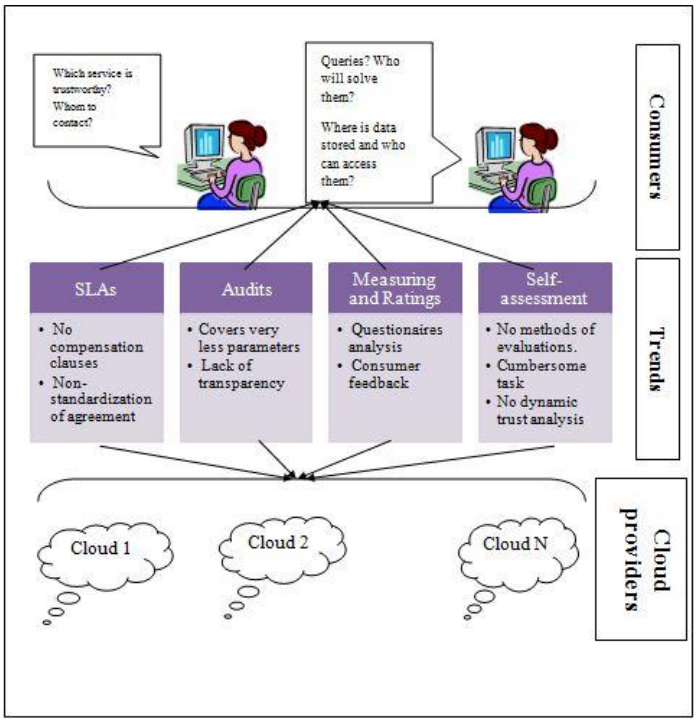


Figure 3.1 *Scope of Improvement in Trust Establishment*

3.2 RELATED WORK ON TRUST IN CLOUD COMPUTING

Zhang et al. (2018) [8]. This paper defines a new model of trust and its co-related algorithm which can overheads of trust management which can further improve intrusion detection of nodes which is based on domain ability. Further, the portioning of nodes helps in decreasing the trust overheads in terms of trust storage and computation. This paper further successfully elaborates the domain and cross domain sliding-windows which can store values of trusts. Afterwards, an algorithm has been designed which will help to compute the values for nodes and a filter is adopted to eliminate malicious trust evaluations and malicious nodes from a domain. Mukalel et al. (2018) [9]. The author proposed a middleware naming Trust Management Middleware (TMM), a framework to select a

trustworthy service within a Cloud. This TMM performs the task of trustworthy service selection process with the help of subjective assessment from users and objective selection from service monitors. Further, they have proposed a new covariance algorithm that can determine the credibility of user feedback. The generated outcome generally demonstrates that how the proposed framework helps in improving the accuracy of trust values. Wang et al.(2015)[10]. The author has proposed a reputation measurement which is light weight approach. It works in two stages such as trust vector and calculation of reputation. In case of trust vector, it implements a classical model to detect feedback scores based on insecurity. In case of reputation calculation, the reputation score is detected by utilizing fuzzy logic and at last, all the calculated values are stored in reputation storage. Fang et al. (2015) [11]. This paper has proposed a structure which is distrust in nature and is useful for recommender systems. It also includes personal and some impersonal features of trust mechanisms. This paper has identified some personal features such as truthfulness, probability, capability etc. whereas impersonal aspects are fully dependent on liveliness of the network. They have also developed regression models logically and using this, they can further calculate trust and distrust values. Marudhadevi et al. 2014 [14] have proposed a trust model namely Trust Mining Model (TMM) to identify trustworthy service of Cloud. This is a model in which user can decide whether to continue or suspend the present services of the service provider. Authors have made use of rough sets and Bayesian inference to calculate the overall trust value of the model .The proposed model have three modules naming a SLA manager, Trust manager and performance monitor. Here, SLA manager is held responsible for dealing with negotiation part among service provider and Cloud consumer. He communicates with trust manager and updated the trust value before the whole agreement is signed. After negotiation of services, they initiate observing services to generate grades if trust through module namely performance monitor. It excludes various parameters such as rejected number of services, network bandwidth and reliability. Users can give feedback on this if they feel any problem. At last, current user feedback and trust value are saved in evidence base along with name of customers for future reference. Rizvi, S et al. (2014) [15] have implemented an Objective Trust Model i.e. feedback system created by the third party auditor in fig 6. In this system, three Cloud components are used namely Cloud service user, Cloud service provider and third party auditor. Cloud service providers are ranked based on trust values evaluated by the model. Also, third part auditor used assessment results and feedback to generate final trust values for each and every Cloud service provider. All these evaluations are done under Cloud Security Alliance (CSA). After the evaluation is done, the final score for each service offered by the CSP is stored in database. The score is rated from 1 to 10 (where 1 represents lowest and 10 represent highest score). Abha et al. (2013) [12] proposed a mechanism that uses AES algorithm that helps in maintaining confidentiality and security of data. In this, the cloud service provider helps in keeping database and its application for users on any remote server and provides independence of accessing them. Also, Cloud Service Provider will now not provide various encryption techniques to each and every user at each level. Jingweiet al. (2013) [13] have illustrated some numerous methods of trust such as SLA reputation, Trust as a Service (TAAS) and transparency. In this Cloud Transparency, the Cloud provider generates self-assessment by two methods i.e. “Consensus Assessments Initiative Questionnaire (CAIQ) or a Cloud Controls Matrix”. The major drawback of implementing this model is untruthful feedback provider who can manipulate generated values of received feedback. Here, TAAS model brings third-party professionals into the picture. This Cloud trust authority introduces a single end for maintaining cloud services security from various providers. The major loophole lies in the form of establishing a trust between users and its trust brokers [21]. Talal et al (2013) [18] have designed and successfully implemented a Cloud Armor, a trust management framework (based on reputation) which is used to deliver the Trust as a Service (TaaS). This framework comprises of several modules such as: A new protocol is being used to deliver the trust of feedback and preserve its user’s privacy. Additionally, an adaptive model (reliable in nature) is used to compute feedbacks helps in securing services from wicked users and finally identifies the honest Cloud services. Dehuaet al (2012) [16] have suggested Trust mechanism based on recommendation on service oriented computing (TRSC). In this, both direct recommendations, trust is used. Web portal is the modes of communication where Cloud providers register their services and user can enlists their requirements and get recommendation results. There are basically three components of TRSC i.e. trust computing, information manager and Cloud services. Here, trust computing provides trust values. When the user uses services, then it provides trust values, after suing services, user can rate the services. If rating is not given by the user then by default rating is generated by the system. Information manager is responsible for managing trust assessment between Cloud user and rating generated.

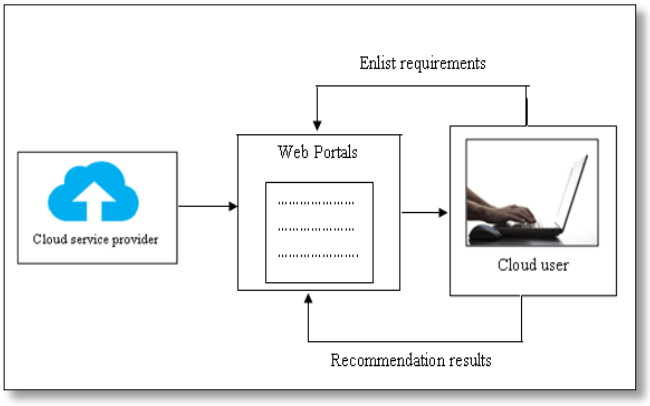


Figure 3.2: *TRSC*

All the trust values are saved in trust and recovery repository. Cloud service provider observes service of Cloud and categorize into different groups and all are kept in Cloud service repository. Talal et al (2011) [17] have reported the several techniques to identify the fake ratings from mischievous users. The techniques used by them are: Credibility proof protocol (CPP) to maintain the privacy of the cloud service users. Also, feedback density is calculated which is used to handle the feedback agreement issues. In addition, false users are detected that generates false test results.

3.3 Existing Trust Models in Cloud Computing

The values of trust can increase of decrease as per the latest experiences and opinions received from external sources or users. These values of trust are computed by using techniques. All the techniques and mechanisms used to calculate trust are collectively known as Trust models. Broadly speaking, a Trust model is a term that can implement mechanisms to generate values of trust so that a user can select a Cloud service as per his requirement. This paper analyzes and categorizes various models of trust on the basis of their approaches to calculate values of trust. The diagram 3.3 depicts an overview of trust models that will be explained after this section.

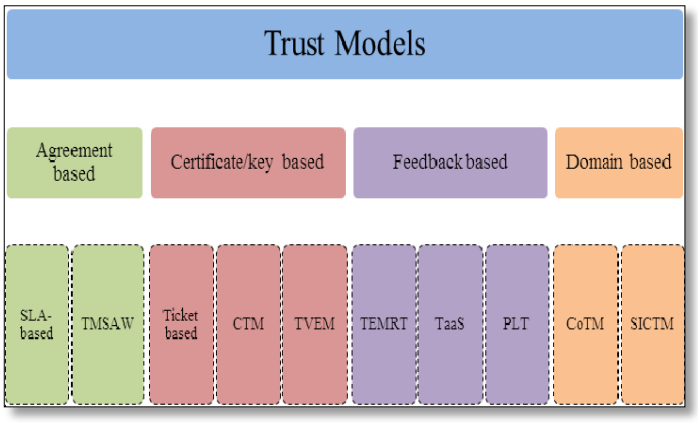


Figure 3.3: *Trust Models classification in Cloud Computing*

3.3.1 Agreement based trust model:
This kind of trust model comes under the category of various agreements that falls under CSP and Cloud service consumers. It works in two different steps. In step I, the consumer lists his various security, privacy and quality of service requirements that are required by him. It is further forwarded for the negotiation. In step II, the trust evaluation model forwards the request to monitoring module for the establishment of trust among both the parties [19, 20]. The following figure 3.4 helps in better understanding of agreement based trust model.

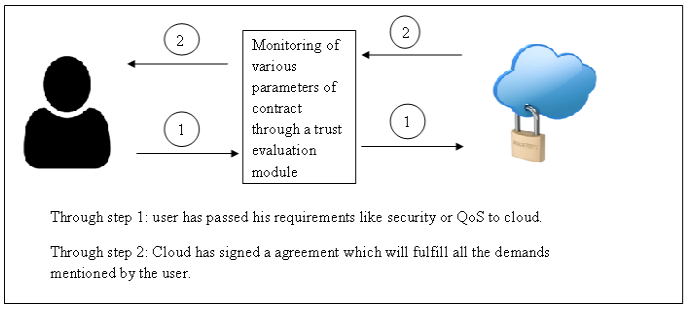


Figure 3.4*: Agreement based Trust model*

Furthermore, Models under Agreement based trust model can be categorized into two types which have been discussed below:

3.3.1.1 SLA – based trust model: It is based on strong SLAs established between the two parties which include various QoS parameters as specified by the consumer. The major components of STM are: Cloud services directory, SLA-agent and Cloud consumer module. The trust evaluation process is done in three steps: firstly, the customers select the CSPs as per his requirement. Secondly, SLA agent is responsible for dividing the customers as per their needs. Here, SLA agent generates a report that will depict reliable and trustworthy CSPs for the users [19]. Lastly, a trust management module is used to calculate final trust model on basis of which consumer will finally select the CSPs.

3.3.1.2 TMSAW: This trust cloud is based on service policies (SP). In this, various service policies are negotiated among the consumers [20]. The idea is to provide two levels of trust layers i.e. internal trust level and contracted trust level. Firstly, if all the operations are under the control of an organization, then internal trust level is established. This trust level is achieved by implementing identity and key management via TPM. Another contract level trust comes into existence when trust is established by negotiating security and QoS requirements.

3.3.2 Certificate/key based trust model
This kind of trust model includes three components are used i.e. certificates, trust tickets (TTs) and keys. Here, certificates are used for software, infrastructure and platform services and are considered one of the practical solution for establishing trust [21]. In this, the certificates are issued by standardized bodies only. Tickets are used to construct integrity and confidentiality and to gain more confidence of customers. This category includes the concept of Trusted Platform Module (TPM) in which data is transferred to the Cloud in an encrypted form from the user [22]. After this, TPM also provides the configuration in an encrypted form to the users.
Following are some of the trust models that falls under the category of certificate/key based trust model:

3.3.2.1 Ticket Based trust model: Mahbub et al. [21] have proposed a new model TTM. When the date is shared between Data Owner (DO) and users, then DO makes use of secret keys to encrypt the data. Here, along with encrypted data, a reference list is also shared that displays user-id, data-id, access rights etc. A new user is registered itself with DO by providing credentials and a reference list is updated along. After this, all the details including expiration time, required credentials and capability list are then forwarded to the new user after registration.

3.3.2.2 Certification-based trust model (CTM): In this customers specify what type of concerns needs to be assessed in a particular Cloud services. The discovery framework Two functions are performed by CTM i.e. certificate comparison and dynamic service composition. Various automated reasoning techniques are used to contrast various certificates of services offered by CSPs. Finally, a list is generated by discovery framework which discovers whether all the evaluated services match with the requirements or not.

3.3.2.3 TVEM-based trust model: This type of trust model contains a TVEM model and a VTN. At initial stages, Data Owner generates a TVEM factory i.e. TF which helps in managing the whole TVEM module, VTN master key, some associated certificates and various trusted environment keys

(TEKs) [23]. After this key creation, TF starts its managing process. Additionally, data is encrypted by using TEK and a VTN is created between a Data Owner and a Cloud Service Provider. Here, TVEM starts measuring of core root values of trust continuously. At the same time, VECR (Virtual Environment Configuration Register) is used for evaluating the dual root of trust [28].

3.3.3 Feedback-based trust models: This category collects various feedbacks and opinions of the users to calculate trust. In this, service registry module is used in which all CSPs are made to register. After the final collection of feedback, a trust score is computed [23, 24]. The first kind of its model is the 'Trust evaluation model based on response time' (TEMRT). It is used to collect the feedback based on response time as offered by Cloud services which finally contributes towards trust evaluation. Another important trust models that is considered under this category is Prepositional Logic Term i.e. PLT in which feedback is collected from different sources and is considered to be key element for trust evaluation. [25]

3.3.3.1 Trust Evaluation Model based on Response Time (TEMRT): This trust model [23] contains three modules or components i.e. trust formulation module, a service monitor and a trust evaluation module. The evaluation of trust is done in two steps. Firstly, the initial stage trust is collected on the basis of various parameters i.e. Quality of Service, mean arrival time and total number of virtual severs available for service. Secondly, next stage feedback is collected on the basis of further two parameters i.e. positive response time feedback (PRT) and negative response time feedback (NRT). In PRT, response time (in case of CSP) is greater than as required threshold value whereas in case of NRT, the value is quite less than as required threshold value. Furthermore, a service monitor module is held responsible for keeping track of the various Cloud services offered to the consumers, and generates continuous feedback to the trust evaluation module. If the services are not provided in accordance with the consumers' needs, then this module informs the trust evaluation module and it contributes to negative values of trust.

3.3.3.2 Trust as a Service model (TaaS):TaaS has a major component which is known as registryservice component. In addition to this, there are three layers in TaaS i.e. Cloud Service Provider (CSP) layer, Trust Management Service Layer (TMSL) layer and a Cloud consumer layer. The CSP layer provides Infrastructure or Platform services to their users or customers. The customers or end users uses the various services of the Cloud by sending appropriate queries and revert with their feedbacks to TMSL. Finally, the TMSL collects customer feedback, compute them and generates a certain value of trust for a specific CSP. A history record is maintained along this whole process that includes ID of CSP, ID of customer giving feedback and some credible feedback reports. Two techniques such as Cloud consumer's capability and majority consensus are used to distinguish between accurate and malicious feedback submitted by Cloud service consumer [24].

3.3.3.3 Prepositional Logic Terms model (PLT): This trust assessment model contains the accompanying modules: (I) Consensus Assessments Initiative Questionnaire (CAIQ) motor, (ii) enlistment administrator (RM), (iii) trust semantic motor (TSE), (iv) Trust calculation motor, (v) trust supervisor (TMg) (vi) and trust update motor (TUE). The procedure of trust definition is performed in three primary advances: (I) the TMg module gives an interface that is utilized by the Cloud clients to indicate their security and QoS prerequisites to assess redid trust score for the CSPs. (ii) Every single trust an incentive for the individual ascribe is consolidated to give a general redid trust score to the client. (iii) The TSE arranges various developments of the PLT that speak to the dependable conduct of CSP as far as explicit properties. (iv) Finally, the TUE is utilized to refresh the trust esteems and gather the criticism from different assets as conclusions that are separated to dispose of any sort of spam or futile data. [25]

3.3.4 Domain-based trust models: Very few models under this category are implemented in Cloud environment. Maximum of them are in use in grid computing. The root idea behind is to divide Cloud into autonomous domains. In this, a particular entity will compute the trust value for another entity. Later on, the trust value is checked from Direct Trust Table. If the value is not found in DTV, then recommended values of trust from other sources are considered. Two types of trust models are categorized under this category i.e. Collaborative Trust Model (CoTM) [26] and 'Security and interoperability centered trust model' (SICTM) [27].

3.3.4.1 Collaborative Trust Model (CoTM):This model [26] separates the Cloud into different self-sufficient areas to such an extent that each CSP keeps up a trust table that keeps the trust estimation of the considerable number of buyers that have collaborated with it inside the space. Each area contains a space operator keeping up three principle tables that incorporate the area inside trust table (DITT), space outside trust table (DOTT) and hazard worth table. DTV in the trust table is increased or decremented as per the achievement or disappointment of exchange history with that particular CSP. On the off chance that no exchange history is available in the trust table for a particular CSP, at that point either the trust an incentive in DITT (inside area) or DOTT (between space) is utilized, contingent on the area of the CSP.

3.3.4.2 Security and Interoperability Centered Trust Model (SICTM): The two noteworthy entities; Cloud clients and CSPs, are separated in the proposed model [27], where every area incorporates the assets that have a place with a similar supplier. The Cloud clients have 'Clients trust tables' and the CSPs keep up 'Space trust tables' that incorporates the area name, administration type, trust degree and age time. Each space incorporates an area trust operator that deals with the trust tables to store the required characteristics for participation between the Cloud suppliers. At first, when a client needs to assess the trust an incentive for a certain CSP, it coordinates the required area name and administration type in the nearby client trust table. The client can begin the exchanges just if the trust worth is more prominent than the characterized edge, generally the procedure is suspended. On the off chance that there is no an incentive in the immediate trust table, at that point the suggested trust score will be utilized.

3.4 PARAMETERS OF TRUST IN CLOUD COMPUTING/ TRUST CALCULATION

Till now, we were discussing about trust. But how this trust is calculated and what are the various parameters on which trust values can be generated. For making decisions in many service environments, TR models have been proven successful. These trust models considers some key experiences to generate trust value i.e. interaction experience, behavioral or technical observations or aspects for selecting entities which as trustworthy in nature. All the mentioned aspects and related parameters have their appropriate importance to conceive while choosing any service providers (trustworthy) in Cloud marketplaces. Cloud services are implemented in massively complex distributed systems. The value of trust can be calculated by considering various parameters and trust models into an account. To accomplish this task, number of trust models needs to be studied as all the parameters cannot be considered in all the models. Many authors have successfully studied trust models and defined trust as a "the belief generated by the users so that other entities can act safely and rely on them". Additionally, trust can be collectively known as multiple attributes such as availability, reliability, security, Quality of services, competence in context of various environments.

Availability (AV):

Availability can be considered as a computative degree in a system or any other module that can be kept in active state and can be used when required [9]. Availability can be measured in terms of mean of total failures and mean time to repair. However, when a job is submitted to any cloud resource, the resource can be said in unavailable state in any of the following situations:

- A part of service of the resource is denied to the user
- The resource is shut down
- The resource is too busy to process the job request.

Assuming R1, R2 … Rm are the cloud resources. For each k = 1, 2 … m, let N_k denote the number of jobs submitted to cloud resource $R_k$over a period T. Out of N_k jobs submitted to R_k, let A_k denote the number of jobs accepted by the resource R_k over the period T.

Availability of the resource R_k (AV) = A_k/ N_k

Reliability (RE):

Reliability is a pivotal component of trust [7]. It is defined as the capability of system or its component to do functions under some specific time period. Finally, reliability can be concluded as a measure of jobs those were successful by the Cloud resource.

Reliability of the resource R_k (RE) = C_k/ A_k

where

C_k denotes the number of jobs completed successfully by R_k

A_k denotes the number of jobs accepted by the Cloud resource.

Data Integrity (DI):

The broader term of data integrity includes security, privacy and accuracy (with precision values) of the data. Additionally, there is a tendency of data loss due to poor network availability.

Data Integrity of the resource R_k (DI) = D_k/ C_k

where

D_k denotes the number of jobs preserved by R_k.

C_k denotes the number of jobs completed successfully by R_k

Now, next section will focus on the concept of fault tolerance in Cloud Computing.

4. FAULT TOLERANCE IN CLOUD COMPUTING

A fault tolerant framework is one that is outlined and worked to keep working notwithstanding when parts of it are inaccessible, or "down." The framework may not really be running at 100% with full efficiency and with all administrations accessible, yet the thought is to keep the framework running at a sensible and usable level [29]. A case of a fault tolerant framework would be a web application made up of numerous administrations (database, storing and application code) that keep running on various servers, for example, you may discover in a miniaturized scale benefit design. Because of an equipment fault, the storing server has turned out to be inaccessible and is disconnected. Rather than the entire application going disconnected, the framework could keep running at a lesser limit by offering highlights that don't require the reserve. When outlining a fault tolerant framework, it's imperative to organize your administrations. You can centre on key administrations, for example, the database, which is an imperative piece of your framework as it controls numerous parts. Another piece of the outline organize is to consider what you would need to happen if a key administration went down. For instance, assume you run a discussion site that gives clients a chance to sign in and post. On the off chance that your validation benefit goes down and a client can't sign in, you can at present give the discussion in a "read just" mode with the goal that clients can in any case discover data and the effect on ease of use is insignificant.

4.1 TYPES OF FAULTS

In general environment, the failures can be temporary or permanent in nature depends upon the hardware or software issues. Sometimes, operator errors can also takes place from server point of view. Also, unintentionally failure can be induced to reduce the throughout and response time of the system.

However in Cloud environment, there are two basic types of failures i.e. Byzantine and crash failures. In byzantine failure, the various components of system fail on rotation basis and in an unpredictable manner. The system may or may not process the request incorrectly and produce undesirable results. In crash failures, the system may crash and will stop completely from functioning or will remain inactive during such failures. As it was discussed earlier in this paper, that the Cloud computing is divided into several and if a fault is detected in any of the layer, then thisFor example, failure in PaaS may generate errors in software services offered by SaaS. Additionally, if a failure occurs in any physical hardware i.e. IaaS, then this may affect negatively on PaaS and SaaS layers. This indicates that failure impact of hardware is quite high as compare to software failure.

4.2 FAULT TOLERANT TECHNIQUES

There are numerous kinds of faults that can hinder a smooth functioning of cloud computing. Based on its classification, two types of fault tolerance techniques can be detected as per the various levels of Cloud computing. Following figure 4.1 represents the major types of techniques of fault tolerance.

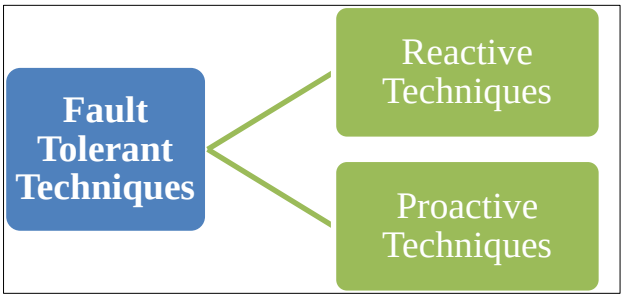


Figure 4.1: *Fault Tolerant Techniques*

4.2.1 Reactive fault tolerance

In this kind of fault tolerance, the policies help in reduction of failures that occurs on any applications being in execution mode. It further states some other 3R methods such as Restart, Replay and Retry. Some of its further divisions can be seen in figure: Checkpointing/Restart: This method is best implemented for long running applications. Hereby, a checkpoint is saved in between any running application. At the time of failure, a restart can be done again from the last saved checkpoint.

Replication: In this, various replicas of the current running module is being run on various other resources [30, 31]. This is being implemented using tools like HAProxy, Hadoop and Amazon etc.

Job Migration: In this, if a failure occurs, a task can be easily and immediately migrated on any other available machine. For example, it can be implemented using HAProxy.

Retry: This is the best method used at the detection of fault. It further retries on the task which is failed on the same Cloud resource.

Task resubmission: In this, at the time of detection of fault, it is resubmitted to either to the same resource or to other available resource [32].

Rescue overflow: This is also one of the widely used techniques. In this, the workflow will continue its execution until and unless the machine completely stops its execution.

4.2.2 Proactive fault tolerance

This technique generally helps in avoiding its recovery from faults and failures and helps in timely exchange of faulted components in the system. Some of its further techniques have been explained below:

Software Rejuvenation: This technique believes in regular rebooting of the system. It is designed in such a way that the system can reboots after some set periods.

Self-healing: In this, failures of various application instances can be handled while they are running on various virtual machines.

Preemptive Migration: It relies on continuous monitored loop system which is constantly checked and analyzed by the other systems available in the environment.

4.3 RELATED WORK ON FAULT TOLERANCE IN CLOUD COMPUTING

Lee et al. (2015) [33]. This paper proposes a fault tolerant and recuperation framework called FRAS framework (Fault tolerant and Recuperation Agent System). This is a specialist based framework comprising of four sorts of operators. Recuperation operator performs move back recuperation after event of disappointment. Data specialist speculation area learning and data amid a disappointment free activity. Facilitator controls the correspondence amongst specialists and refuse gathering operator performs trash accumulation of information. Operator recuperation calculation is proposed to keep up a steady condition of a framework and anticipate domino impact. Dipankar et al. (2014) [34]. This paper has successfully proposed and implemented a framework that uses research on different model of compliance that can be used as an interface by the companies to test risk factors and various compliances. The tool that was developed generates as SLA document for organization which helps in detecting the correct required services from cloud service provider in order to certify the compliance. Sagar et al. (2014) [35]. It has proposed a mechanism based on fault tolerant that can handle various failures by migration of various machines from a failure machine to a new location. This has given rise to the concept of Virtual Data Centers (VDCs), where allocation can be done on virtual machines. Also, multiple Virtual Data Centers can be hosted on data center physically, by using appropriate allocation algorithms. Nuygen et al. (2013) [36]. This paper has elaborated one of the vast challenges in case of fault tolerance, that directly pinpoints the faulty components in a system. There is a Black Box online fault localization system which is further known as F-Chain that can successfully points out the faulty components immediately when any anomaly is detected in a system. This model is implemented in case of Infrastructure as a service and does not depend on any previously detected or unseen anomalies. Further, an integrated fault localization scheme has been introduced that helps in achieving higher pinpointing inaccuracy which considers fault propagation patterns and other functional dependencies. Sun et al. (2013) [37]. This paper has elaborated a dynamic adaptation to internal failure procedure (DAFT) that is engaged around the norms and semantics of cloud adaptation to non-critical failure. An examination on connection between various disappointment rates and two diverse adaptations to non-critical failure procedures, registration and replication has been completed. A dynamic versatile model has been worked by joining the two adaptations to non-critical failure models which builds the serviceability.

4.4 EXISTING FAULT TOLERANCE MODELS

In this, various existing models of fault tolerance have been described. In some other architectures, more than above explained techniques have been used.

4.4.1 HAProxy - Fault Tolerance model: It is first kind of reactive architecture. In this, the techniques of job migration and replication are used. In its operation, there are two server machines and one HAProxy on it which is on monitoring mode. In this, one server machine has its replica on other server machine. If the first server machine fails, the second server machine will back up the operation of its own to deal with fault occurred. Here, HAProxy is responsible for

maintaining these various tasks [38]. Now, in this, it is possible that both the servers are in active mode but there is no possibility that both the servers will be on passive mode.

4.4.2 Byzantine Fault Tolerance (BFT Cloud) model: In this architecture, replication policy is used. In BFT cloud [39], whenever a request arrives in a system, it is forwarded to different nodes. Here, any one of the node is considered as primary node and other as secondary nodes. Further, the requests are executed on all the nodes. If the outputs of primary as well as all the secondary nodes are same then the output is correct and same is forwarded to the module which requested it.
If the output is not correct from any of the node, then that node is known as faulty node and recovery operations are performed in the updating stage. Also, if the faulty node is primary node then the primary version is replaced with the new node and if it secondary node then it is replaced with any one of replicas available in the updating stage (as shown in figure 4.2).

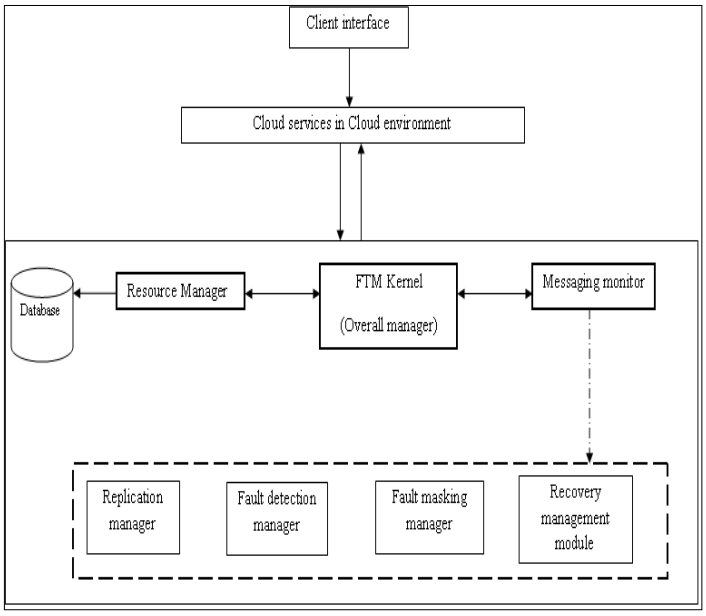


***Figure 4.2**: BFT Cloud*

4.4.3 Fault Tolerance Manager (FTM) fault tolerance model: It is a kind of reactive architecture which implements three methods of fault tolerance i.e. replication, checkpoint/restart and job migration. Here, FTM stands for Fault tolerance manager[40]. FTM works with the help of three modules i.e. FTM kernel, resource manager and messaging monitor. Figure 4.3 depicts the same. Firstly, FTM kernel acts as manager and decides what kind of fault is detected. Secondly, resource manager provides login information in form of database under the authority of virtual machines in the cloud. Lastly, the messaging monitor is primarily managed by FTM manager and has further two tasks. Firstly, it checks the messages among the replica modules and secondly it monitors the messages were exchanged among different modules. The module has four further sub modules i.e. replication manager (generates replica from desired node), fault detection/predication manager (predict fault), fault masking manager (decides the fault recovery methods) and recovery management module (perusal of fault recovery policy). In addition to this, two communication modules are responsible for linking layer of fault tolerant and Cloud management.

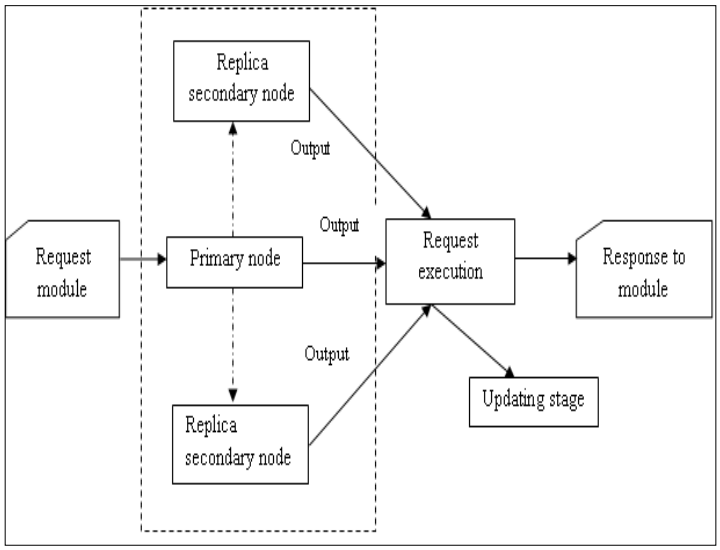


***Figure 4.3:** FTM Fault Tolerance Model*

4.4.4 Application Fault Tolerance in Real Time Cloud Computing (AFTRC) fault tolerance model: Most of the Cloud applications are based on RTHPC i.e. Real Time High performance Cloud Computing [41]. Therefore, the systems are sensitive to response time at the specific period of time. That means, if a response is generated after specific time period then its output will not have any value
or importance. Also, these fault tolerance techniques helps in increasing the system's response time and displays the output generation in appropriate time.

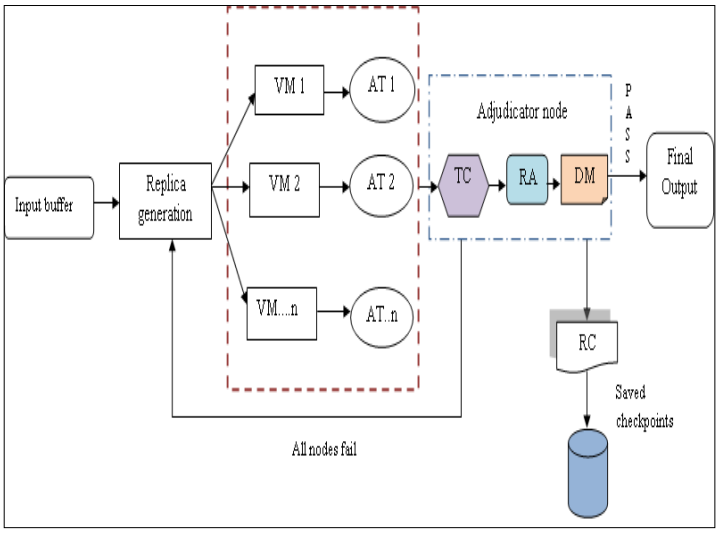


***Figure 4.4**: AFTRC – Fault Tolerance Model*

For this, AFTRC is used. By receiving input from input buffer, replicas are generated on various virtual machines. Next stage includes job migration policy, transfer and migration of works to different virtual machines (as shown in figure 4.4). Here, a special module is used i.e. Acceptance Test (AT). In this validity of the output is passed to TC module (Time Checker) and if the output is found invalid then the exception signal is sent through TC. Also, RA module (Reliability assessor) will evaluate the reliability of virtual machines outputs on the basis of AT and TC. Additionally, RA module will set a maximum and minimum limit of reliability and if the reliability is less than minimum, it will remove that specific node. Finally, DM module (Decision maker) will generate the final output based on reliability of computing cycle. RC module (Recovery cache) is a store that keeps the record of various checkpoints so that if any last saved checkpoint needs to be performed. The three nodes i.e. TC, RA and DM are collectively known as adjudicator node.

4.4.5 GOSSIP Architecture: This architecture was introduced to improve the performance of Byzantine architecture. It is based on reactive method which uses replication policy to detect fault and creates fault tolerance capability in Cloud environment [42]. In this architecture, decision vector is used. At every network operation stage, each node selects a neighbor node and any two nodes are required to update their decision vector altogether. The size of this decision vector depends on the number of neighboring nodes it has. The faulty node will generate a different output each time at the time of output generation and thus conflicts the value of decision vector each time. This architecture improves the fault detection to 50%. For example, if there is 2k+1 nodes are available on the environment, then k can be the number of faulty nodes that can be detected.

4.4.5 GOSSIP Architecture: This architecture was introduced to improve the performance of Byzantine architecture. It is based on reactive method which uses replication policy to detect fault and creates fault tolerance capability in Cloud environment [42]. In this architecture, decision vector is used. At every network operation stage, each node selects a neighbor node and any two nodes are required to update their decision vector altogether. The size of this decision vector depends on the number of neighboring nodes it has. The faulty node will generate a different output each time at the time of output generation and thus conflicts the value of decision vector each time. This architecture improves the fault detection to 50%. For example, if there is 2k+1 nodes are available on the environment, then k can be the number of faulty nodes that can be detected.

4.4.6 Process Level Redundant (PLR) Fault Tolerance model: This architecture is for real time high performance Cloud systems. The major reason for this name is that it can create redundancy in the process. It has also embedded MPI in an interesting form. This model has five modules for its working. The first module is fault predictor that helps in predicting the type of fault which has been detected. The next module is PLR Controller Daemon which is helpful in monitoring the message passing from one node to another. The third module is Fault Tolerance Policy which is responsible for selecting the type of policy while dealing with a fault. The fourth module is Fault Tolerance Dream which initially sends a monitoring message to PLR. In the end, Checkpoint/restart module stores the check points on the various neighboring nodes to remove the failure points and release the resource to use saved check points. The policies which were designed for creating and increasing the fault tolerance capability. Increase the response time and output production. Also, its users have claim atthis architecture reduces the response time and shows a better performance compared to other architectures [43].

4.5 FAULT TOLERANCE PARAMETERS

Different parameters were taken into consideration while dealing with many existing fault-tolerant methods in the Cloud computation.

Performance: this is used to verify the efficiency of the system. It needs to be improved at a reasonable cost e.g. by reducing response time while keeping acceptable delays.

Response Time: It is the total amount of time taken to respond to particular algorithm or request. This parameter should have minimum value.

Throughput: This denotes the number of tasks that has completed its execution. It should be on the higher side to improve its efficiency.

Reliability: This parameter aims at providing correct or acceptable result within a bounded environment.

Availability: This parameter helps in ensuring that particular item will be available in a satisfactorily state at a given point of time under some bounded conditions. As the reliability increases, the availability factor also increases.

Usability: This measures the extent to which a product can be used to obtain some specific goals with effectiveness, efficiency and a level of satisfaction.

Overhead Associated: It determines the levels of overheads detected while implementing any fault tolerant algorithm. It is composed of overhead due to interprocess communications. This parameter should have minimum value so that algorithm

can work in an efficient mode.

5. COMPARTITIVE ANALYSIS OF TRUST AND FAULT TOLERANCE:

The following table 5.1 depicts the compartitive analysis on the basis of two challenges i.e. trust and fault tolerance. This compartitive analysis has been done by taking procedures and related challenges into consideration.

Trust/ Fault Tolerance	Model Type	Model Name	Process involved	Challenges
Trust	Agreement based	STM	Trust value is generated on the basis of SLAs	Lack of QoS transparency. Limited support to functional features.
		TMSAW	Service policies are negotiated that provides two levels of trust i.e. internal and contractual.	Lack of QoS transparency. Not flexible in nature.
Trust	Certificate-based trust	TTM	Data is encrypted with the help of secret keys.	Lack of QoS transparency. Does not support ease of applicability.
		CTM	Certificates are issued that evaluates as the trust values.	Lack of QoS transparency. Does not support ease of applicability
		TVEM	Data is encrypted using TEKs and VTN.	Data is not available all the time.
Trust	Feedback Trust category	TEMRT	Evaluates trust on basis of QoS.	Process Execution control is not fully supported.
		TaaS	Trust feedback is calculated which generates the credible values of trust.	Data is not available all the time.

Trust	Domain Trust category	CoTM	Domain wise trust is generated and is maintained in DITT.	Data Confidentiality is not fully supported. Process Execution control is not fully supported.
		SICTM	Trust value is generated with the help of CSPs and various domains and is stored in trust tables.	Lack of QoS transparency. Data availability is not fully supported.
Fault Tolerance	Reactive Fault Tolerant Method	FTM	Virtualization is used which is based on reliability, availability and on-demand service.	Lack of in-depth analysis of FT services.
Fault Tolerance	Proactive Fault Tolerant Method	BFT	Ranking is done for some components.	Latency part is not covered.
Fault Tolerance	Adaptive Fault Tolerant Method	AFTRC	Backward recovery through checkpointing.	Node is removed which is incompetent in nature.
Fault Tolerance	Reactive Fault Tolerant Method	HAProxy	Virtualization concept is used.	Errors were generated at the time of endurance testing.

Table 5.1 *Comparative Analysis of Trust and Fault Tolerance*

6. CONCLUSION & FUTURE SCOPE

In this paper, we have done in-depth study on various Cloud-based trust models as well as Fault Tolerance models in Cloud computing. Trust model in Cloud computing is the most in-demand mechanism that helps in building secured communication in Cloud environment. The objective of this review is to introduce mechanism which can further help in selecting a CSP which will be reliable as well as fault tolerant in nature. Additionally, Adaptation to non-critical failure techniques become an integral factor the minute a shortcoming enters the framework limits. So hypothetically adaptations to internal failure systems are utilized to foresee these failures and make a proper move before failures really happen. Also, we have tried best to review about various faults tolerant models. In the last, we have done a detailed comparison analysis on the basis of various parameters. Moreover, till date no trust model has been proposed which inculcates the feature of fault tolerance with its credibility efficiency.The extension of this paper will propose a fault tolerance trust mechanism which will reduce the challenges those where highlighted in compartitive analysis table. Also, various mobile agents can be incorporated that can ease out the work by contributing in calculating the efficient trust values of a fault tolerant mechanism in Cloud environment.

REFERENCES

[1] Yildiz, Mehmet, Abawajy, Jemal, Ercan, Tuncay and Bernoth, Andrew, A layered security approach for cloud computing infrastructure, Proceedings of the 10th International Symposium on the Pervasive Systems, Algorithms and Networks, pp. 763-767.

[2] Habib, S. M., Hauke, S., Ries, S., &Mühlhäuser, M. (2012). Trust as a facilitator in cloud computing: a survey. Journal of Cloud Computing: Advances, Systems and Applications, 1(1),

[3] Chong, S. K., Abawajy, J., Ahmad, M., & Hamid, I. R. A. (2014). Enhancing trust management in cloud environment.Procedia-Social and Behavioral Sciences, 129, 314-321.

[4] Cheraghlou, M. N., Khadem-Zadeh, A., &Haghparast, M. (2016). A survey of fault tolerance architecture in cloud computing. Journal of Network and Computer Applications, 61, 81-92.

[5] S. Perez, “In Cloud We Trust?” ReadWriteWeb, Jan. 2009; www.readwriteweb.com/enterprise/2009/01/incloud-we-trust.php.

[6] Fujitsu Research Institute (2010) Personal data in the cloud: A global survey of consumer attitudes. Technical Report, Fujitsu Research Institute.

[7] Uusitalo I, Karppinen K, Juhola A, Savola R (2010) Trust and cloud services - an interview study. In: Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on, p. 712–720.

[8] Zhang, P., Kong, Y., & Zhou, M. (2018). A Domain Partition-Based Trust Model for Unreliable

Clouds.IEEE Transactions on Information Forensics and Security, 13(9), 2167-2178.
[9] Smithamol, M. B., &Rajeswari, S. (2018). TMM: Trust Management Middleware for Cloud Service Selection by Prioritization. Journal of Network and Systems Management, 1-27.
[10] Wang, S., Guo, Y., Li, Y., & Hsu, C. H. (2018). Cultural distance for service composition in cyber-physical-social systems.Future Generation Computer Systems.
[11] Fang, H., Guo, G., & Zhang, J. (2015). Multi-faceted trust and distrust prediction for recommender systems. Decision Support Systems, 71, 37-47.
[12] Sachdev, A., &Bhansali, M. (2013). Enhancing cloud computing security using AES algorithm.International Journal of Computer Applications, 67(9).
[13] Huang, J., &Nicol, D. M. (2013). Trust mechanisms for cloud computing. Journal of Cloud Computing: Advances, Systems and Applications, 2(1), 9.
[14] Marudhadevi, D., Dhatchayani, V. N., &Sriram, V. S. (2014). A trust evaluation model for cloud computing using service level agreement. The Computer Journal, 58(10), 2225-2232.
[15] Rizvi, S., Ryoo, J., Liu, Y., Zazworsky, D., &Cappeta, A. (2014, May). A centralized trust model approach for cloud computing. In 2014 23rd Wireless and Optical Communication Conference (WOCC) (pp. 1-6). IEEE.
[16] Kong, D., &Zhai, Y. (2012, November). Trust based recommendation system in service-oriented cloud computing. In 2012 International Conference on Cloud and Service Computing (pp. 176-179).IEEE.
[17] Noor, T. H., & Sheng, Q. Z. (2011, December). Credibility-based trust management for services in cloud environments.In International Conference on Service-Oriented Computing (pp. 328-343).Springer, Berlin, Heidelberg.
[18] Noor, T. H., Sheng, Q. Z., Ngu, A. H., Alfazi, A., & Law, J. (2013, October). Cloud armor: a platform for credibility-based trust management of cloud services. In Proceedings of the 22nd ACM International Conference on Information & Knowledge Management (pp. 2509-2512).ACM.
[19] Alhamad, M., Dillon, T. and Chang, E. (2010) SLA-Based Trust Model for Cloud Computing. 13th Int. Conf. Network-BasedInformation Systems (NBiS), Takayama, Japan, September 14– 16, pp. 321–324. IEEE Computer Society, Los Vaqueros.
[20] Sato, H., Kanai, A. and Tanimoto, S. (2010) A Cloud Trust Model in a Security Aware Cloud. 10th IEEE Int. Symp.Applications and the Internet (SAINT), Korea, July 19–23, pp. 121–124. IEEE, New York, USA.
[21] Ahmed, M. and Xiang, Y. (2011) Trust Ticket Deployment: A Notion of a Data Owner'S Trust in Cloud Computing. 10th Int. Conf. Trust, Security and Privacy in Computing and Communications (TrustCom), China, November 16–18, pp. 111–117.IEEE, New York, USA.
[22] Krautheim, F.J., Phatak, D.S. and Sherman, A.T. (2010) Introducing the Trusted Virtual Environment Module: A New Mechanism for Rooting Trust in Cloud Computing. 3rd Int. Conf. Trust and Trustworthy Computing, Germany, June 21–23, pp. 211–227.Springer, Berlin.
[23] Firdhous, M., Ghazali, O. and Hassan, S. (2011) A Trust Computing Mechanism for Cloud Computing. Proc. ITU, The Fully Networked Human?-Innovations for Future Networks and Services (K-2011), Cape Town, December 12–14, pp. 1–7. IEEE, New York, USA.
[24] Noor, T.H. and Sheng, Q.Z. (2011) Trust as a Service: A Framework for Trust Management in Cloud Environments. Web Information System Engineering–WISE, Sydney, Australia, October 13–14, pp. 314–321. Springer, Berlin.
[25] Habib, S.M., Ries, S. and Muhlhauser, M. (2011) Towards a TrustManagement System for Cloud Computing. 10th Int. Conf. Trust, Security and Privacy in Computing and Communications (TrustCom), Changsha, November 16–18, pp. 933–939. IEEE, New York, USA.
[26] Yang, Z., Qiao, L., Liu, C., Yang, C. and Wan, G. (2010) A Collaborative Trust Model of Firewall-Through Based on Cloud
[27] Computing. 14th Int. Conf. Computer Supported Cooperative Work in Design (CSCWD), China, April 14–16, pp. 329–334. IEEE, New York, USA.
[28] Li, W. and Ping, L. (2009) Trust Model to Enhance Security and Interoperability of Cloud Environment. 1st Int. Conf. Cloud Computing (CloudCom), China, December 1–4, pp. 69–79. Springer, Berlin.
[29] Cheraghlou, M. N., Khadem-Zadeh, A., &Haghparast, M. (2016). A survey of fault tolerance architecture in cloud computing. Journal of Network and Computer Applications, 61, 81-92.
[30] Rodero, F. Guim, J. Corbalan, "Evaluation of coordinated Grid scheduling strategies." In High Performance Computing and Communications, 2009.HPCC'09. 11th IEEE International Conference on, pp. 1-10. IEEE, 2009.
[31] T. Chalermarrewong, T. Achalakul, and S. C. W. See."The design of a fault management framework for cloud."In Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2012 9th International Conference on, pp. 1-4.IEEE, 2012.
[32] K. Ganga., and S. Karthik. "A fault tolerant approach in scientific workflow systems based on cloud computing." In Pattern Recognition, Informatics and Medical Engineering (PRIME), 2013 International Conference on, pp. 387-390. IEEE, 2013.
[33] Lee, H., Chung, K., Chin, S., Lee, J., Lee, D., Park, S., & Yu, H. (2005). A resource management and fault tolerance services in grid computing. Journal of Parallel and Distributed Computing, 65(11), 1305-1317.
[34] Dasgupta, D., &Rahman, M. (2014, October).A framework for estimating security coverage for cloud service insurance.In Proceedings of the Seventh Annual Workshop on Cyber Security and Information Intelligence Research (p. 40).ACM.
[35] Joshi, S. C., &Sivalingam, K. M. (2014). Fault tolerance mechanisms for virtual data center architectures. Photonic Network Communications, 28(2), 154-164.
[36] Nguyen, H., Shen, Z., Tan, Y., &Gu, X. (2013, July). Fchain: Toward black-box online fault localization for cloud systems. In Distributed Computing Systems

(ICDCS), 2013 IEEE 33rd International Conference on (pp. 21-30).IEEE.

[37] Sun, D., Chang, G., Miao, C., & Wang, X. (2013). Analyzing, modeling and evaluating dynamic adaptive fault tolerance strategies in cloud computing environments. The Journal of Supercomputing, 66(1), 193-228.

[38] Bala, A., &Chana, I. (2012). Fault tolerance-challenges, techniques and implementation in cloud computing. International Journal of Computer Science Issues (IJCSI), 9(1), 288.

[39] Zhang, Y., Zheng, Z., &Lyu, M. R. (2011, July). BFTCloud: A byzantine fault tolerance framework for voluntary-resource cloud computing. In 2011 IEEE 4th International Conference on Cloud Computing (pp. 444-451).IEEE.

[40] Kaur, J., &Kinger, S. (2013). Analysis of different techniques used for fault tolerance. IJCSIT Int. J. Comput.Technol, 4, 737-741.

[41] Lakshmi, S. S. (2013). Fault tolerance in cloud computing. Int. J. Eng. Sci. Res. IJESR, 4, 1285-1288.

[42] Lim, J., Chung, K. S., Gil, J. M., Suh, T., & Yu, H. (2013, February). An unstructured termination detection algorithm using gossip in cloud computing environments.In International Conference on Architecture of Computing Systems (pp. 1-12).Springer, Berlin, Heidelberg.

[43] Egwutuoha, I. P., Chen, S., Levy, D., &Selic, B. (2012, May). A fault tolerance framework for high performance computing in cloud.In 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012) (pp. 709 - 710).IEEE.