Soft Computing (2019) 23:13591–13602
https://doi.org/10.1007/s00500-019-03896-6

METHODOLOGIES AND APPLICATION

An efficient method for fault tolerance in cloud environment using encryption and classification

Vipul Gupta[1] · Bikram Pal Kaur[2] · Surender Jangra[3]

Published online: 17 April 2019

Abstract
Cloud computing may be defined as management and provision of resources, software, application and information as services over the cloud which are dynamically scalable. Fault tolerance includes all the techniques necessary for robustness and dependability. The main advantages of using fault tolerance in cloud computing include failure recovery, lower costs and improved standards in performance. Even though the benefits are immeasurable, the element of risk on user applications due to failure remains a major drawback. So our suggested technique utilizes the effective fault tolerance method with the encryption algorithm. To improve the security of the recommended technique, triple-DES encryption algorithm is employed before the data transmission. For the transmission of encrypted data, the implemented method selects the minimum fault tolerance node. So the recommended technique utilizes the effective classification technique. Here, improved support vector machine (ISVM) classifier is used to classify the nodes based on its feature value and the content similarity each node. The proposed ISVM helps in predicting the faults if available, earlier before it occurs. The various parameters considered in our proposed system are accuracy, service reliability and availability. In the proposed method, the accuracy value of the fault tolerance is 79% which is better than in the existing method. The proposed method will be implemented in JAVA with CloudSim.

Keywords Cloud computing · Fault tolerance · Triple DES · Support vector machine · Accuracy · Service · Availability

1 Introduction

Cloud computing is a large-scale distributed computing paradigm driven by economies of scale, in which a pool of abstracted, virtualized, dynamically scalable, highly available and configurable and reconfigurable computing resources can be rapidly provisioned and released with minimal management effort in the data centers (Sun et al. 2013). Cloud computing has become popular; it offers to use several advantages such as cost saving, flexibility, scalability and agility, over traditional computing models (Muhra et al. 2014). Cloud computing is widely used in different IT services, including, but not limited to, parallel computing, visualization, network storage technologies, load balance, utility computing, service-oriented, etc. (He et al. 2016; Choi et al. 2015). Cloud applications usually involve the greatest number of components; it is still too costly to provide alternative components for all the cloud components. The cloud computing techniques provide low-cost and fast computing resources with advanced capacity architecture (Zhang et al. 2015). To reduce the cost so as to develop highly reliable cloud applications within a limited budget, a small set of critical components needs to be classified from the cloud applications. In cloud computing, a large number of servers may lead to enhancement of overall node failure probability, so a high failure rate fault tolerance is essential (Liu 2015).

Fault tolerance of cloud computing is dynamic which leads to unexpected system behavior resulting in faults and failures. In order to improve reliability and achieve

Communicated by V. Loia.

✉ Vipul Gupta
vipulgupta10853@gmail.com

[1] Punjab Technical University, Kapurthala, India

[2] Department of Computer Science and Engineering, Chandigarh Engineering College, Landran, Mohali, India

[3] Department of Computer Applications, GTB College, Bhawanigarh (Sangrur), Punjab, India

robustness in cloud computing, failures should be assessed and handled effectively (Yang et al. 2015). Fault tolerance is the property that enables a system to continue operating properly in the event of the failure of some of its component (Menychtas and Konstanteli 2012). It is an essential part of Service Level Objectives (SLOs) in clouds. High fault tolerance issue is one of the major obstacles for opening up a new era of high serviceability cloud computing as fault tolerance plays a key role in ensuring cloud serviceability (Sun et al. 2013). There are two subareas of fault tolerance in computer systems: hardware fault tolerance and software fault tolerance (Karaca 2012). There are three fault-tolerant task clustering methods to enhance existing task clustering approaches in a faulty distributed execution environment. The first method retries failed tasks within a job by extracting them into a new job. The second method dynamically adjusts the granularity or clustering size according to the estimated inter-arrival time of task failures. The third method partitions the clustered jobs into finer jobs and retries them. Fault detection is one of the biggest challenges in making a system fault tolerant. Fault detection methods aim to detect signs of abnormal network operation based on monitoring specific networks performance parameters such as packet loss and delay (Menychtas and Konstanteli 2012).

Limitation of fault tolerance is evaluating the performances of fault tolerance component in comparison with similar ones; a benchmark-based method should be developed to grantee high reliability and availability (Gupta and Banga 2013). The major drawback of this approach is that the fault-free resources which are interconnected with the faulty resource cannot be used. This leads to a reduction in system performance (Latif et al. 2015). Fault recovery, one of the fault tolerance techniques that adopt checkpointing and rollback/roll-forward scheme, can enable a task to recover from an error and resume the processing of the task. That minimizes the effect of the faults or, if possible, provides recovery (Menychtas and Konstanteli 2012; Yang et al. 2009; Zheng et al. 2012; Chen et al. 2012; Abujarad et al. 2015).

2 Related works

Zhu et al. (2016) have first established a real-time workflow fault-tolerant model that improved the standard PB model by including the cloud characteristics. Primarily based on that model, they developed approaches for task shared and communication transmission to ensure faults could be tolerated throughout the workflow execution. At last, they proposed a dynamic fault-tolerant scheduled algorithm, FASTER, for real-time workflows in the virtualized cloud hosting. FASTER had three key characteristics: (1) It implements a backward shifted method to make full use of the idle resources and incorporates task overlapping and VM migration for high resource used, (2) it applied the vertical/horizontal scaling-up way to quickly provision resources for a burst of workflows, and (3) it used the vertical scaling-down scheme to avoid unnecessary and ineffective source changes due to fluctuated workflow recommended. They evaluated FASTER algorithm with synthetic workflows and workflows collected from the real scientific and business applications and compared it with six baseline algorithms.

Availability is one of the most essential requirements in the production system. Keeping a persistent level of high availability in the Infrastructure-as-a-Service (IaaS) cloud computing is a test because of the complexity of service providing. By definition, the availability can be kept up by coupling with the fault tolerance approaches. As of last, many fault tolerance methods have been created; however, few of them adequately consider the fault detection aspect, which is critical to issuing the appropriate recovery actions just in time. Bui et al. (2018) have proposed a rigorous analysis on the nature of failures, and they would like to introduce a method to early identify the faults occurred in the IaaS system. By connecting fuzzy logic algorithm and prediction technique, the proposed approach could provide better performance in terms of accuracy and reaction rate, which subsequently enhances the system reliability.

In cloud computing, resources are dynamically provisioned and sent to users in a transparent manner automatically on demand. Process execution failure is no longer accidental but a common characteristic of cloud computing environment. Recently, a number of intelligent arranging techniques have been used to address task arranging issues in the cloud without much attention to fault tolerance. Abdulhamid et al. (2018) have presented a powerful clustering league championship algorithm (DCLCA) scheduled approaches for fault tolerance awareness to address cloud task performance which would be reflected on the currently available resources and reduced the untimely failure of autonomous tasks. Experimental results showed that approaches produced remarkable fault reduction in task failure as measured in terms of failure rate. That also demonstrated that the DCLCA outperformed the MTCT, MAX–MIN, ant colony optimization and genetic algorithm-based NSGA-II by produced lower makespan with an improvement of 57.8, 53.6, 24.3 and 13.4% in the first scenario and 60.0, 38.9, 31.5 and 31.2% in the second scenario, respectively. Considered the experimental results, DCLCA provides better quality fault tolerance aware scheduled that will help to improve the overall performance of the cloud environment.

As clouds have been implemented widely in various areas, the reliability and availableness of clouds become

the major concern of cloud companies and users. Therefore, fault tolerance in clouds receives a great offer of attention in both industry and academia, especially for real-time applications because of their safety severe nature. Huge amounts of analyzer have been conducted to realize fault tolerance in distributed systems, among which fault-tolerant scheduling plays a significant role. Forever, little amount of analyzers on the fault-tolerant scheduling study the virtualization and the flexibility, two key characteristics of clouds, sufficiently. To address this problem, Wang et al. (2015) have shown a fault-tolerant mechanism which increases the Primary-Backup (PB) model to incorporate the characteristics of clouds. At the same time, they proposed an elastic resource provisioning system in the fault-tolerant framework to increase the resource used. Based on the fault-tolerant system and the flexible resource provisioning mechanism, they designed novel fault-tolerant elastic algorithms for real-time jobs in clouds named FESTAL, aiming at achieving both fault tolerance and high resource utilization in clouds.

The minimizing mean-time-to-failure estimates in cloud computing systems suggest that multimedia applications working on such environments must be able to mitigate an improving quantity of core failures at runtime. Anarado and Andreopoulos (2016) have proposed a fresh roll-forward failure mitigation approach for integer sum-of-product computations, with special emphasis on high-performance general matrix multiplication (GEMM) and convolution/cross-correlation (CONV) routines. It based on the production of unnecessary results within the numerical representation of the results via the use of numerical packing. That varies from all existing roll-forward solutions that required a separate set of checksum (or duplicate) outcomes. The imposes 37.5% reduction in the maximum output bit width supported in comparison with integer GEMM or CONV realizations performed on 32-bit integer representations. However, that bit width decreased similarly to the one made due to the checksum elements of traditional roll-forward methods, especially for instances where multiple core failures must be mitigated.

Structural health monitoring (SHM) systems are structured for setups (e.g., bridges, buildings) to monitor their functions and health status. Cellular sensor networks (WSNs) have become an enabling technology for SHM applications that are definitely more prevalent and more quickly deployable than classical wired networks. However, SHM brings new challenges to WSNs: engineering-driven optimal deployment, a huge amount of data, complex computing and so on. Bhuiyan et al. (2015) have resolved two important challenges: sensor deployment and decentralized processing. They proposed a solution to deploy wireless sensors at strategic locations to achieve the best estimation of structural health (e.g., damage) by using the widely used wired sensor system deployment approach from civil/structural engineering. They found that faults (caused by communication errors, unstable online connectivity, sensor faults, etc.) in such an implemented WSN greatly affect the performance of SHM. To make the WSN resistant to the faults, they present an approach, called FTSHM (fault tolerance in SHM), to repair the WSN to guarantee a specific level of fault tolerance. FTSHM searches the restoring points in clusters in a distributed manner and places a couple of backup sensors at those points in such a way that still satisfies the engineering requirements. FTSHM also includes an SHM algorithm suite for decentralized computing in the energy-constrained WSN, with the goal of assuring that the WSN for SHM remains linked in the event of a fault, thus prolonging the WSN lifetime under connectivity and data delivery constraints. They have demonstrated the benefits of FTSHM through intensive simulations and real experimental options on a physical structure.

Infrastructure-as-a-service clouds are becoming broadly adopted. However, resource posting and multi-tenancy have made performance anomalies a top concern for users. On time debugging those anomalies is paramount for decreasing the performance penalty for users. Unfortunately, this debugging often takes a long time because of the inherent complexity and sharing nature of cloud infrastructures. When software activates a performance anomaly, it is important to tell apart between errors with a global impact and faults with a local impact as the diagnosis and recovery steps for faults with a global impact or local impact are quite different. Dean et al. (2016) have presented PerfCompass, an internet performance anomaly fault debugging tool that could quantify whether a production-run performance anomaly has a global impact or local impact. PerfCompass might use this information to suggest the main cause as either another fault (e.g., environment based) or an internal fault (e.g., software bugs). Since PerfCompass could identify top damaged system calls to provide useful diagnostic hints for comprehensive performance debugging, PerfCompass does indeed require source code or runtime uses instrumentation, which makes it useful for production systems.

Deng et al. (2017a) have presented a novel intelligent diagnosis method. That was used to diagnose the faults of the motor bearing. The vibration signal was decomposed into a set of intrinsic mode functions (IMFs) by using empirical mode decomposition method. The fuzzy information entropy values of IMFs were calculated to reveal the intrinsic characteristics of the vibration signal and considered as feature vectors. Then, the diversity mutation strategy, neighborhood mutation strategy, learning factor strategy and inertia weight strategy for basic particle swarm optimization (PSO) algorithm were used to generate

an improved PSO algorithm. The improved PSO algorithm was used to optimize the parameters of least squares support vector machines (LS-SVM) in order to construct an optimal LS-SVM classifier that was used to classify the fault. Finally, they also presented a fault diagnosis method was fully evaluated by experiments and comparative studies for motor bearing.

Deng et al. (2018) have presented a new motor bearing fault diagnosis method based on integrating EWT, fuzzy entropy and support vector machine (SVM), called EWTFSF. Their presented method, a novel signal processing method called EWT, was used to decompose vibration signal into multiple components in order to extract a series of amplitude-modulated–frequency-modulated (AM–FM) components with supporting Fourier spectrum under an orthogonal basis. Then, fuzzy entropy was utilized to evaluate the complexity of vibration signal, reflect complexity changes in intrinsic oscillation and compute the fuzzy entropy values of AM–FM components that were regarded as the inputs of SVM model to train and construct a SVM classifier for fulfilling fault pattern recognition. Finally, the effectiveness of their presented method was validated by using the simulated signal and real motor bearing vibration signals.

Deng et al. (2017b) have presented a genetic and ant colony adaptive collaborative optimization (MGACACO) algorithm for solving complex optimization problems. Their presented MGACACO algorithm makes use of the exploration capability of GA and stochastic capability of ACO algorithm. Their presented MGACACO algorithm, the multi-population strategy was used to realize the information exchange and cooperation among the various populations. The chaotic optimization method was used to overcome long search time, avoid falling into the local extremum and improve the search accuracy. The adaptive control parameter was used to make relatively uniform pheromone distribution and effectively solve the contradiction between expanding search and finding optimal solution. The collaborative strategy was used to dynamically balance the global ability and local search ability and improve the convergence speed. Finally, various scales TSP are selected to verify the effectiveness of their presented MGACACO algorithm.

Deng et al. (2017c) have presented an adaptive particle swarm optimization (DOADAPO) algorithm based on making full use of the advantages of alpha-stable distribution and dynamic fractional calculus was deeply presented in their article. The dynamic fractional calculus with memory characteristic was used to reflect the trajectory information of particle updating in order to improve the convergence speed. The alpha-stable distribution theory was used to replace the uniform distribution in order to escape from the local minima in a certain probability and improve the global search ability. Next, the DOADAPO algorithm was used to solve the constructed multi-objective optimization model of gate assignment in order to fast and effectively assign the gates to different flights in different times. Finally, the actual flight data in one domestic airport were used to verify the effectiveness of their presented method.

Zhao et al. (2018) have presented a damage degree identification method based on high-order difference mathematical morphology gradient spectrum entropy (HMGSEDI). The main aim of their method was to solve the issue that fault signal of rolling bearings was weak and difficult to be quantitatively measured. In the HMGSEDI method, on the basis of mathematical morphology gradient spectrum and spectrum entropy, the changing scale influence of structure elements to damage degree identification was thoroughly analyzed to determine that optimal scale range. The high-order difference mathematical morphology gradient spectrum entropy was then defined in order to quantitatively describe the fault damage degree of bearing. The discrimination concept of fault damage degree was defined to quantitatively describe the difference between the high-order differential mathematical entropy and the general mathematical morphology entropy in order to present a fault damage degree identification method. The vibration signal of motors under no-load and load states was used to testify the effectiveness of their presented HMGSEDI method.

Zhao et al. (2016) have presented a new fault feature extraction method, called the EDOMFE method based on integrating ensemble empirical mode decomposition (EEMD), mode selection, and multi-scale fuzzy entropy is proposed to accurately diagnose fault in those article. The EEMD method was used to decompose the vibration signal into a series of intrinsic mode functions (IMFs) with a different physical significance. The correlation coefficient analysis method was used to evaluate and determine three improved IMFs, and those were close to the original signal. The multi-scale fuzzy entropy with the ability of effectively distinguishing the complexity of different signals was used to evaluate the entropy values of the selected three IMFs in order to form a feature vector with the complexity measure, that was regarded as the inputs of the support vector machine (SVM) model for training and constructing a SVM classifier (EOMSMFD based on EDOMFE and SVM) for fulfilling fault pattern recognition. Finally, the effectiveness of their presented method was validated by real bearing vibration signals of the motor with different loads and fault severities.

The purpose of question classification (QC) is to assign a question to an appropriate category from the set of predefined categories that constitute question taxonomy. Selected question features are able to significantly improve

the performance of QC. However, feature extraction, particularly syntax feature extraction, has a high computational cost. To maintain or enhance performance without syntax features, Liu et al. (2018) have presented hybrid approach to semantic feature extraction and lexical feature extraction. Those features were generated by improved information gain and sequential pattern mining methods, respectively. Selected features were then fed into classifiers for questions classification.

With the expanding request and advantages of cloud computing foundation, ongoing computing can be performed on cloud framework. A continuous framework can exploit serious computing capacities and scalable virtualized condition of cloud computing to execute ongoing undertakings. In the greater part of the continuous cloud applications, preparing is done on remote cloud computing hubs. So there are more odds of mistakes, because of the undetermined latency and free authority over computing hub. On the opposite side, the majority of the continuous frameworks is likewise well-being basic and ought to be very dependable. So there is an expanded necessity for fault tolerance to accomplish reliability for the ongoing computing on cloud foundation. This motivates us to do the research on fault tolerance in cloud environment.

3 Proposed methodology

Cloud computing has become a widely used system in order to obtain computing resources in a cost-effective manner. Even though the benefits are immeasurable, the element of risk on user applications due to failure remains a major drawback. In order to perform the functions of any system to the required level, fault tolerance is a major concern. So our suggested technique utilizes the effective fault tolerance method with an encryption algorithm. The main aim of the proposed method is to select the minimum fault tolerance node for transmission and secure the data before transmission. At first user authentication is verified and then the data are stored in the data center. In order to improve the security of the recommended technique, the encryption algorithm is utilized before uploading the data to the data center. For encryption, triple-DES (TDES) algorithm is employed. After that, we store the encrypted data to the cloud data center. For the transmission of encrypted data, we have to select the minimum fault tolerance node. So here we used improved support vector machine (ISVM) classifier for selecting the nodes. Here, the node classification is based on its feature value and the content similarity each node. Here, the traditional support vector machine algorithm is improved by means of the kernel function. The classification used here helps us in predicting the faults, if available earlier before it occurs. The novelty of the suggested method is encrypting the data before storing the cloud data center and selects the minimum fault tolerance node with the help of ISVM algorithm. The overall process is described in the further section (Fig. 1).

In this paper, we have designed a new approach in order to enhance the fault tolerance problems in cloud computing. At first user authentication is verified and then the input data are encrypted by means of triple-DES algorithm. Next, we select the node for transmission with minimum fault tolerance using improved support vector machine algorithm. The detail explanation of each process is described as follows.

3.1 User authentication

The primary step in our proposed system is the user authentication wherein various users are assigned with individual authentication data. For utilizing the cloud, each user has to login with their authentication details and can store or edit data. The authentication is provided in order to present more security to personal information by avoiding unauthorized access to the stored information of a particular user. In our proposed system, we have employed secure storage of data using TDES algorithm as it provides more security than normal storage process. Once these stages are done, the final stage is the transmission of data. For the transmission of data, we have to select particular nodes so that proper transmission of data can be done. For the selection of nodes, we utilized improved SVM algorithm for classification. The step-by-step procedure of encryption and classification is illustrated in further section.

3.2 Triple-DES algorithm

Triple DES applies the data encryption standard (DES) cipher algorithm three times to each data block. In general, TDES was introduced to have three keys having a key length of 168 bits (three 56-bit DES keys). When it was discovered that a 56-bit key of DES is not enough to protect from attacks, TDES was chosen as a simple way to enlarge the keyspace without a need to switch to a new algorithm. Triple DES simply extends the key size of DES by applying the algorithm three times in succession with three different keys. Using this process, the input data are encrypted for secure transmission.

In triple DES, there are three keying options in data encryption standards, such as:

(a) All keys being independent
(b) Key1 and key2 being independent keys
(c) All three keys being identical

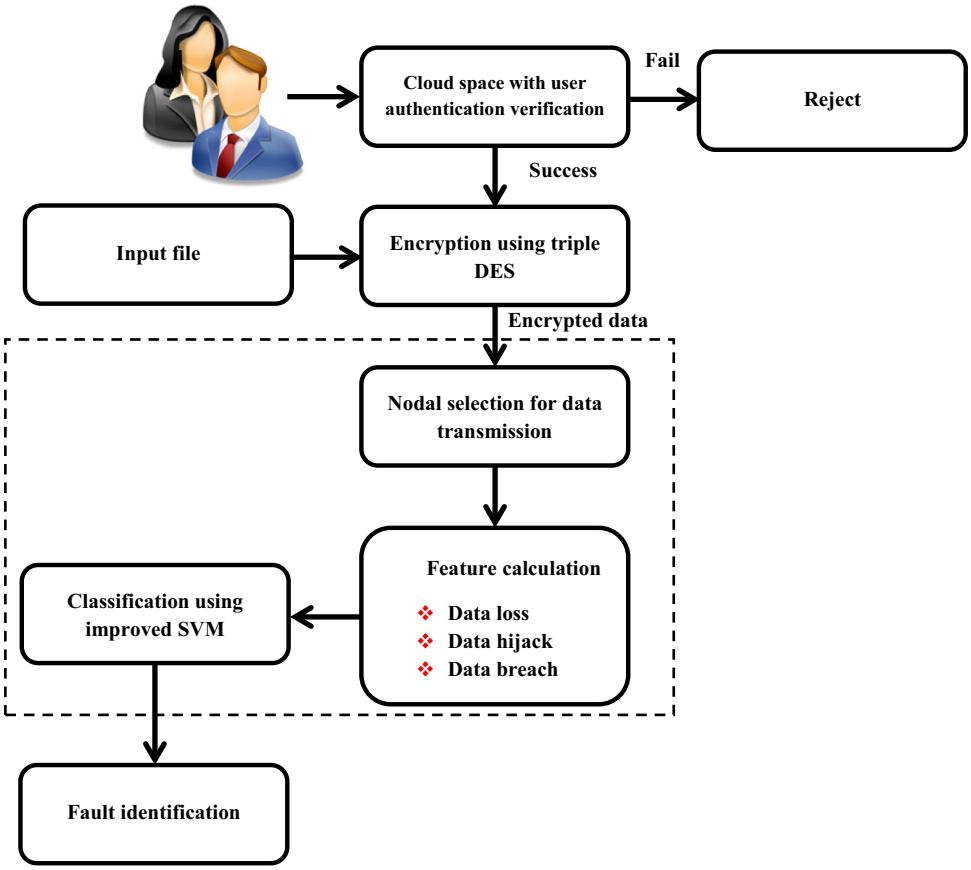

Fig. 1 Proposed flow diagram

Triplcmmon DES cipher. It receives a secret 168-bit key, which is divided into three 56-bit keys.

- Encryption using the first secret key
- Decryption using the second secret key
- Encryption using the third secret key

The process of triple-DES algorithm is shown in Fig. 2. The process of encryption is as follows:

1. Encrypt the data using DES algorithm with the help of the first key.
2. Now, decrypt the output generated from the first step using the DES algorithm with the help of the second key.
3. Finally, encrypt the output of the second step using DES algorithm with the help of the third key.

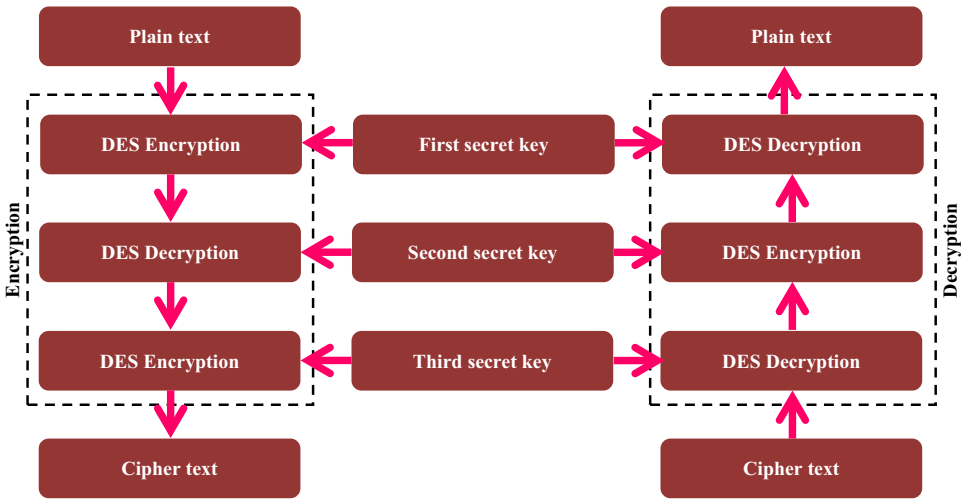

Fig. 2 TDES process for data encryption

$$C = E_3(D_2(E_1(d))) \quad (1)$$

The decryption process of any ciphertext that was encrypted using triple-DES algorithm is the reverse of the encryption process

1. Decrypt the ciphertext using DES algorithm with the help of the third key.
2. Now, encrypt the output generated from the first step using the DES algorithm with the help of the second key.
3. Finally, decrypt the output of the second step using DES algorithm with the help of the first key.

$$d = D_1(E_2(D_{31}(C))) \quad (2)$$

Triple DES is advantageous because it has a significantly sized key length, which is longer than most key lengths affiliated with other encryption modes. DES algorithm was replaced by the advanced encryption standard, and triple DES is now considered to be obsolete. It derives from single DES but the technique is used in triplicate and involves three subkeys and key padding when necessary. Keys must be increased to 64 bits in length known for its compatibility, and flexibility can easily be converted for triple-DES inclusion. Based on the above process, the proposed method encrypts the data, and the next process is to select the minimum fault tolerance node for transmission.

3.3 Classification using ISVM

Support vector machine (SVM) is one of the well-known techniques for its optimization solution. Despite its good theoretic foundations and generalization performance, SVM is not suitable for classification of large data sets since SVM needs to solve the quadratic programming problem (QP) in order to find a separation hyperplane, which causes an intensive computational complexity. Hence, the improved SVM technique is proposed in our paper to apply the classification process for selecting the minimum fault tolerance node. At first, the feature values are calculated from each node. Then, the selected features from the above process will be used as the training input for the improved support vector machine (ISVM) classification. As the ISVM is trained with the input features, it is easy to identify the minimum fault tolerance node.

3.3.1 Feature extraction

For selecting the minimum fault-free node, the suggested technique extracts the features from each input node. Here, we are considering three features such as data breaches, data loss and data hijacking. The description of the three features is illustrated as follows:

1. Data breaches

When a virtual machine is able to access the data from another virtual machine or organization's sensitive internal data fall into the hands of their competitors, a data breach occurs. The side-channel attacks are valid attack vectors; they attack timing information to extract private cryptographic keys being used in other virtual machines on the same physical server. Multitenant cloud service database is not properly designed, and a flaw in one client's application could allow attacker access not only to that client's data but every other client's data as well.

Countermeasure To encrypt your data to reduce the impact of a data breach, but if you lose your encryption key, you will lose your data as well.

2. Data loss

Data loss can happen in the cloud when data get into wrong hands while transferring or be lost due to the hard drive failure. A CSP could accidentally delete the data; an attacker might modify the data. Any accidental such as a fire or earthquake could lead to the permanent loss of customer's data.

Countermeasure The provider must take adequate measures to backup data.

3. Data hijacking

Credentials and passwords are often reused to access our data in the cloud by which an attacker gaining access to our data can manipulate and change the data.

Countermeasures: Organizations should look to prohibit the sharing of data credentials between users and services and leverage strong two-factor authentication techniques where possible.

After calculating the feature value form the node, these features are fed to the input for ISVM classifier. There are two vital phases in the ISVM process such as the training phase and the testing phase.

Training phase The output of features of each node is furnished as the input of the training phase. The input function furnishes the set of values which cannot be separated. Almost all the potential segregations of the point set are realized by means of a hyperplane. But for classifying the input node, it is difficult to find the separation hyperplane. In order to reduce the quadratic programming problem (QP) on finding the separation hyperplane, a Lagrange multiplier is introduced to the SVM classifier. In the Lagrange configuration, it is possible to locate the separation of the hyperplane normal vector through the dissimilar kernel function. The common version of the kernel function is furnished as follows.

$$K(a,b) = \phi(a)^{\mathrm{T}}\phi(b) \tag{3}$$

The efficiency of the SVM invariably relies on the choice of the kernel. In the event of the feature space being linearly inseparable, it has to be mapped into a superior dimensional space by means of the radial basis function kernel, in order that the issue emerges as linearly separable. Moreover, the combination of any two or more kernel functions is competent to yield superlative precision than that obtained by employing any single kernel function.

3.3.2 Improved support vector machine (ISVM)

Improved SVM utilizes the kernel function to classify the node without the use of hyperplanes. The kernel functions generally used in practice are of linear, quadratic, polynomial, sigmoid and RBF (radial basis function) kernel function. Given underneath are the expressions for the different kernel functions.

$$\text{For Linear Kernel: } L_K(a,b) = a^{\mathrm{T}}b + c \tag{4}$$

where a, b indicate the internal products in linear kernel and c represents constant.

$$\text{For Quadratic Kernel: } Q_K(a,b) = 1 - \frac{\|a-b\|^2}{\|a-b\|^2 + c} \tag{5}$$

where a, b indicate the vectors of the polynomial kernel function in the input orientation.

$$\text{For Polynomial Kernel: } P_K(a,b) = \left(\gamma a^{\mathrm{T}}b + c\right)^{\in}, \quad \gamma > 0 \tag{6}$$

$$\text{For Sigmoid Kernel: } S_K(a,b) = \tanh\left(\gamma a^{\mathrm{T}}b + c\right), \quad \gamma > 0 \tag{7}$$

In our innovative technique, linear and quadratic kernel functions were combined and the average of two functions is used here to separate the minimum fault tolerance node. The innovative work employs the integration of two kernel functions given as follows:

$$\mathrm{Avg}(a,b) = \frac{(a^{\mathrm{T}}b + c) + \left(1 - \frac{\|a-b\|^2}{\|a-b\|^2 + c}\right)}{2} \tag{8}$$

The above-mentioned equation can be again represented as:

$$\mathrm{Avg}(a,b) = \frac{(L_K(a,b) + Q_K(a,b))}{2} \tag{9}$$

where $Avg(a,b)$ indicates the incorporated kernel function characterization of the linear and quadratic kernels.

Testing phase The output from the feature selection is furnished as an input to the testing phase and the output indicates the input dataset as normal or abnormal.

With the above-mentioned classification technique, we have been able to achieve a minimum fault-free classification of nodes through which the secure data transmission can be performed.

4 Result and discussion

The proposed method of fault tolerance in the cloud using soft computing technique is implemented in the working platform of JAVA. We have actualized our proposed fault tolerance in the cloud using JAVA (Netbeans 8.2 and jdk 1.8) with CloudSim devices, and a progression of experiments were performed on a PC with Windows 7 Operating system at 2 GHz dual-core computer with 4 GB RAM running a 64-bit version of Windows 2007. For transmission, the fault-free nodes are selected by utilizing our proposed classification algorithm of improved support vector machine. The effectiveness of the proposed system is evaluated by comparing with existing systems.

4.1 Evaluation metrics

4.1.1 Availability

The availability is defined as the ratio of uptime to the sum of the uptime and the downtime.

$$\text{Availability} = \frac{\text{uptime}}{(\text{uptime} + \text{downtime})} \tag{10}$$

4.1.2 Service reliability

The service reliability is defined as the ratio of successful response to the total request with the percentage 100.

$$\text{Service reliability} = \left(\frac{\text{successful response}}{\text{total request}}\right) * 100 \tag{11}$$

4.2 Performance evaluation

The evaluation parameters for our proposed system are encryption and decryption time, upload and download time, availability and service reliability. For various iterations and file size, the corresponding encryption and decryption time, upload and download time, availability and service reliability are noted.

Table 1 shows the upload and download time obtained for different file sizes. If the size is 10, the upload time is 3845 ms and the download time is 1965 ms. Then, if the size is 20, the upload and download time is 4635 ms and 2004 ms, respectively. If the size is 30, the upload time is 6112 ms but the download time is 2156 ms, and if the size

Table 1 Upload and download time for different sizes

File size (kb)	Upload time (ms)	Download time (ms)
10	3845	1965
20	4635	2004
30	6112	2156
40	7562	2536

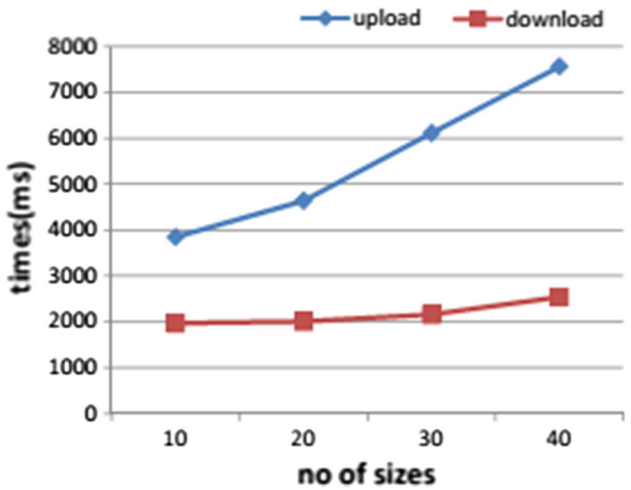

Fig. 3 Upload and download time versus number of sizes

Table 2 Encryption and decryption time for different file sizes

File size (kb)	Encryption time (ms)	Decryption time (ms)
10	1923	1635
20	2452	2111
30	3259	2678
40	4561	3274

is 40, the upload time is 7562 ms and the download time is 2536 ms. The graphical representation of upload and download time is shown in Fig. 3.

The encryption and decryption time of the proposed method for different sizes is given in Table 2. If the size is 10, the encryption time is 1923 ms and the decryption time is 1635 ms. Then, if the size is 20, the encryption and decryption time is 2452 ms and 2111 ms, respectively. If the size is 30, the encryption time is 3259 ms but the decryption time is 2678 ms, and if the size is 40, the encryption time is 4561 ms and the decryption time is 3274 ms. The graphical representation of upload and download time is shown in Fig. 4.

In order to evaluate the fault tolerance performance, the proposed method calculates the availability and service reliability by varying the number of iteration. Table 3

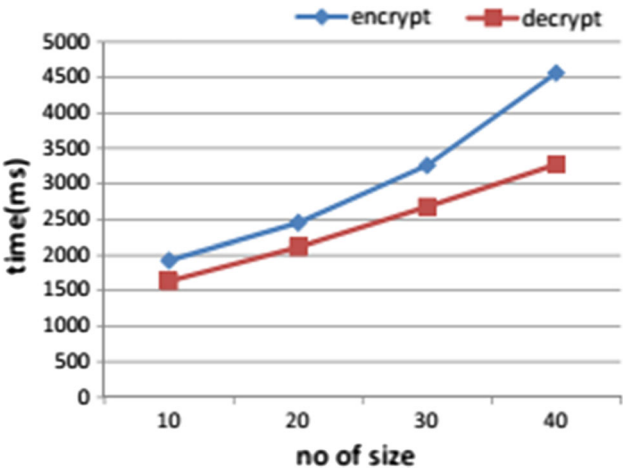

Fig. 4 Encrypt and decrypt time versus number of sizes

Table 3 Availability for different iterations

Iteration	Availability
10	0.661
20	0.698
30	0.739
40	0.748

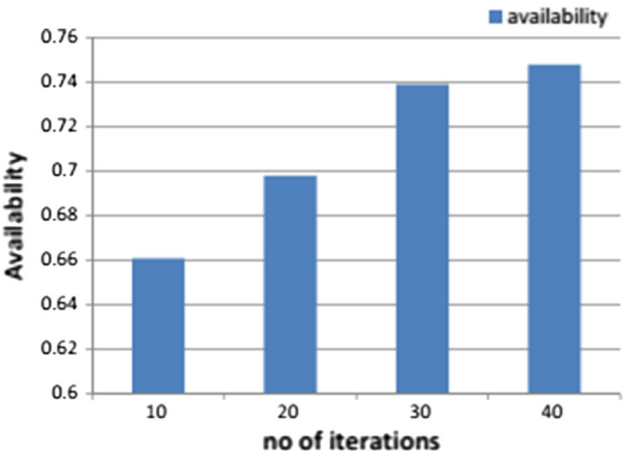

Fig. 5 Graphical representation of the availability

shows the availability performance of the proposed method.

If the iteration is 10, the fault tolerance availability value is 0.661. If the iteration is 20, the value of this availability is 0.698. If the iteration is 30 and 40, here the availability value is 0.739 and 0.748, respectively. The graphical representation of availability is shown in Fig. 5.

Table 4 shows the service reliability of the proposed method by varying the number of iterations.

If the iteration is 10, the service reliability of fault tolerance is 0.78. If the second iteration is 20, the value of

Table 4 Proposed service reliability for different iterations

Iteration	Service reliability
10	0.78
20	0.74
30	0.68
40	0.89

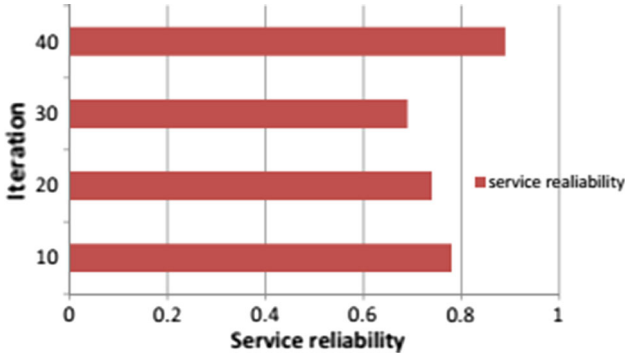

Fig. 6 Graphical representation of service reliability

Table 5 Proposed accuracy for different iterations

Iteration	Accuracy
10	0.72
20	0.76
30	0.78
40	0.80

Table 6 Time complexity analysis of proposed method

File size (kb)	Time complexity (ms)
10	2483
20	3082
30	3949
40	5311

service reliability attains the 0.74. Here, if the iteration becomes 30, the fault tolerance of service reliability reaches the 0.68. If the iteration is 40, the service reliability value attains the 0.89 (Fig. 6).

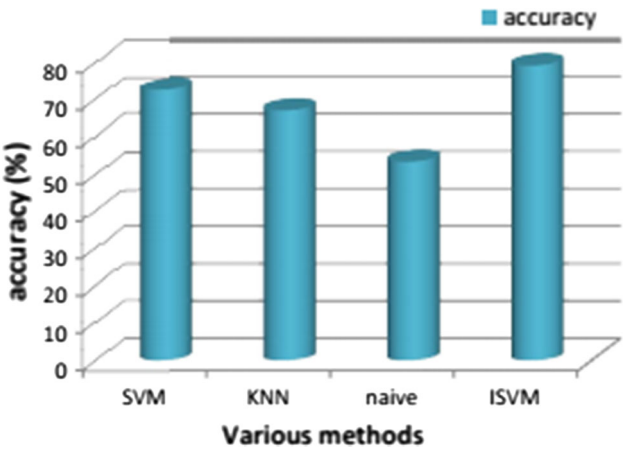

Fig. 7 Comparison result of proposed accuracy

The accuracy value of the proposed technique for different iterations is given in Table 5. If the accuracy value reaches the maximum value of the proposed technique effectively select the fault-free node.

If the iteration is 10, the accuracy value achieves the 72%, and if the iteration is 20, the accuracy value reaches 76%. If the iteration is 30 and 40, the value of accuracy is 78% and 82%, respectively. The accuracy value reaches the maximum value in a different iteration.

The time complexity analysis of the proposed method is given in Table 6. When analyzing the table, the proposed method reaches the time complexity of 2483 ms for 10 kb file size. For 20 kb and 30 kb, the proposed method reaches the time complexity of 3082 ms and 3949 ms, respectively. The proposed time complexity value for 40 kb file size is 5311 ms. In order to prove the effectiveness of the proposed method, the implemented method is compared with the various existing algorithm in the following section.

4.2.1 Comparative analysis

In this section, we explain the efficiency of the fault tolerance in the cloud using encryption and classification technique. To prove the efficiency of the proposed work, we compare the proposed work to existing work. Here, for comparison hybrid PSO–MCS algorithm is utilized.

Table 7 Comparative analysis of the proposed and existing method

File size	Existing method (PSO-MCS)		Proposed method (TDES-ISVM)	
	Upload time (ms)	Download time (ms)	Upload time (ms)	Download time (ms)
10 kb	9254	5424	3845	1965
20 kb	9978	5984	4635	2004

Table 8 Encryption and decryption time for different file sizes

File size	Proposed method TDES		RSA		Blowfish	
	Encryption time (ms)	Decryption time (ms)	Encryption time (ms)	Decryption time (ms)	Encryption time (ms)	Decryption time (ms)
10 kb	1923	1635	1928	1642	1935	1654
20 kb	2452	2111	2461	2118	2473	2125

The comparison result of the proposed technique and the existing technique is given in Table 7. If the file size is 10, the upload time of the existing method is 9254 and the download time is 5424. Here, the upload time of the proposed method is 3845 and the download time is 1965. If the full size is 20, the upload and the download time of the existing method is 9978 and 5984, respectively. Then, the upload time of the suggested method attains the 4635 and the download time reaches 2004. Comparing the existing and proposed methods, the proposed method reaches the value at a minimum time.

Figure 7 shows the plot of the proposed comparison of accuracy value. In the existing method, the accuracy value of support vector machine is 72.4%, and KNN classifier achieves the accuracy value of 66.8%. The accuracy value for the naïve Bayes method achieves 53.1%. In the proposed method, improved support vector machine reaches the accuracy value of 78.8% maximum value when compared with the existing method.

The comparison result of the proposed encryption and decryption time for different file sizes is given in Table 8. For comparison, here we are considering RSA and Blowfish algorithm. If the file size is 10 kb, the existing method of RSA algorithm has the encryption time of 1928 ms and the decryption time of 1642 ms. The encryption time for the existing Blowfish algorithm is 1935 ms and the decryption time is 1654 ms. But the proposed method reaches the encryption and decryption time of 1923 ms and 1635 ms, respectively, which is a minimum value when compared with the existing methods. If the file size is 20 kb, here the existing method of the RSA reaches the encryption time of 2461 ms and the decryption time of 2118 ms and the Blowfish algorithm reaches the encryption and decryption time of 2473 ms and 2125 ms, respectively. Here, the proposed method has the encryption time of 2452 ms and the decryption time of 2111 ms, which is a minimum value when compared with the existing method. From the experimental result, it is clear that the suggested technique attains the minimum encryption and decryption time and achieves maximum accuracy value when compared with the existing algorithm.

5 Conclusion

In a section, the effective technique is used for fault tolerance in a cloud environment. Here, the encryption is done through TDES algorithm and the classification is done through improved support vector machine. The performance of the proposed method was evaluated by the accuracy, availability, service reliability, and encryption and decryption time. Experimental results indicate that the proposed method framework has outperformed by having better accuracy of 79% when compared with the existing method. SVM achieves 72%, KNN classifier attains 69%, and naïve Bayes classifier attains the 53%. For the comparison of encryption and decryption time, the proposed method reaches the minimum time when compared with the existing Blowfish and RSA algorithm. From the result, the proposed method provides minimum fault-free node with more secure transmission when compared with the existing techniques. In future, the researcher will have sufficient opportunities to perform with various encryption and classification techniques for selecting the fault tolerance node with more secure transmission and produce newer heights of excellence in performance.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical statement I confirm that the manuscript has not been submitted to more than one journal for simultaneous consideration. The manuscript has not been published previously (partly or in full) unless the new work concerns an expansion of previous work.

References

Abdulhamid SM, Abd Latiff MS, Madni SHH, Abdullahi M (2018) Fault tolerance aware scheduling technique for cloud computing environment using dynamic clustering algorithm. Neural Comput Appl 29(1):279–293

Abujarad F, Lin Y, Bonakdarpour B, Kulkarni SS (2015) The complexity of automated addition of fault-tolerance without explicit legitimate states. Distrib Comput 28(3):201–219

Anarado I, Andreopoulos Y (2016) Core failure mitigation in integer sum-of-product computations on cloud computing systems. IEEE Trans Multimed 18(4):789–801

Bhuiyan MZA, Wang G, Cao J, Wu J (2015) Deploying wireless sensor networks with fault-tolerance for structural health monitoring. IEEE Trans Comput 64(2):382–395

Bui DM, Huynh-The T, Lee S (2018) Early fault detection in IaaS cloud computing based on fuzzy logic and prediction technique. J Supercomput 74(11):5730–5745

Chen W, da Silva RF, Deelman E, Fahringer T (2012) Dynamic and fault-tolerant clustering for scientific workflows. IEEE Trans Cloud Comput 4(1):49–62

Choi S, Chung K, Yu H (2015) Fault tolerance and QoS scheduling using CAN in mobile social cloud computing. Clust Comput 17(3):911–926

Dean DJ, Nguyen H, Wang P, Gu X, Sailer A, Kochut A (2016) PerfCompass: online performance anomaly fault localization and inference in infrastructure-as-a-service clouds. IEEE Trans Parallel Distrib Syst 27(6):1742–1755

Deng W, Yao R, Zhao H, Yang X, Li G (2017a) A novel intelligent diagnosis method using optimal LS-SVM with improved PSO algorithm. Soft Comput 1–18

Deng W, Zhao H, Zou L, Li G, Yang X, Wu D (2017b) A novel collaborative optimization algorithm in solving complex optimization problems. Soft Comput 21(15):4387–4398

Deng W, Zhao H, Yang X, Xiong J, Sun M, Li B (2017c) Study on an improved adaptive PSO algorithm for solving multi-objective gate assignment. Appl Soft Comput 59:288–302

Deng W, Zhang S, Zhao H, Yang X (2018) A novel fault diagnosis method based on integrating empirical wavelet transform and fuzzy entropy for motor bearing. IEEE Access 6:35042–35056

Gupta P, Banga S (2013) Topic-review of cloud computing in fault tolerant environment with efficient energy consumption. Int J Sci Res Manag (IJSRM) 1(4):251–254

He J, Dong M, Ota K, Fan M, Wang G (2016) NetSecCC: a scalable and fault-tolerant architecture for cloud computing security. Peer-to-Peer Netw Appl 9(1):67–81

Karaca O, Sokullu R (2012) A cross-layer fault tolerance management module for wireless sensor networks. J Zhejiang Univ Sci 13(9):660–673

Latif K, Rahmani A-M, Nigussie E, Seceleanu T, Radetzki M, Tenhunen H (2015) Partial virtual channel sharing: a generic methodology to enhance resource management and fault tolerance in networks-on-chip. J Electron Test 29(3):431–452

Liu D (2015) A fault-tolerant architecture for ROIA in cloud. J Ambient Intell Humaniz Comput 6(5):587–595

Liu Y, Yi X, Chen R, Zhai Z, Gu J (2018) Feature extraction based on information gain and sequential pattern for English question classification. IET Softw

Menychtas A, Konstanteli KG (2012) Fault detection and recovery mechanisms and techniques for service oriented infrastructures. In: Achieving real-time in distributed computing: from grids to clouds, pp 259–274

Muhra A, Vu QH, Asal R, Al Muhairi H, Yeun CY (2014) Lightweight secure storage model with fault-tolerance in cloud environment. Electron Commer Res 14(3):271–291

Sun D, Chang G, Miao C, Wang X (2013) Analyzing, modeling and evaluating dynamic adaptive fault tolerance strategies in cloud computing environments. J Supercomput 66(1):193–228

Wang J, Bao W, Zhu X, Yang LT, Xiang Y (2015) FESTAL: fault-tolerant elastic scheduling algorithm for real-time tasks in virtualized clouds. IEEE Trans Comput 64(9):2545–2558

Yang B, Tan F, Dai Y-S, Guo S (2009) Performance evaluation of cloud service considering fault recovery. Cloud Comput 571–576

Yang C-T, Liu J-C, Hsu C-H, Chou W-L (2015) On improvement of cloud virtual machine availability with virtualization fault tolerance mechanism. J Supercomput 69(3):1103–1122

Zhang W, Xu L, Duan P, Gong W, Lu Q, Yang S (2015) A video cloud platform combing online and offline cloud computing technologies. Pers Ubiquit Comput 19(7):1099–1110

Zhao H, Sun M, Deng W, Yang X (2016) A new feature extraction method based on EEMD and multi-scale fuzzy entropy for motor bearing. Entropy 19(1):14

Zhao H, Yao R, Xu L, Yuan Y, Li G, Deng W (2018) Study on a novel fault damage degree identification method using high-order differential mathematical morphology gradient spectrum entropy. Entropy 20(9):682

Zheng Z, Zhou TC, Lyu MR, King I (2012) Component ranking for fault-tolerant cloud applications. IEEE Trans Serv Comput 5(4):540–550

Zhu X, Wang J, Guo H, Zhu D, Yang LT, Liu L (2016) Fault-tolerant scheduling for real-time scientific workflows with elastic resource provisioning in virtualized clouds. IEEE Trans Parallel Distrib Syst 27(12):3501–3517