Impact Factor (SJIF): 3.632

International Journal of Advance Research in Engineering, Science & Technology

e-ISSN: 2393-9877, p-ISSN: 2394-2444

Volume 3, Issue 6, June-2016

A Self-tuning Failure Detection Scheme in Cloud Environment

Rajesh Biradar[1], Ramesh Ganiger[2]

[1,2]*Department of computer science and engineering, Basaveshwar Engineering College –Bagalkot*

Abstract - Cloud computing is an important solution for providing services dynamically in cloud networks. Services in cloud are virtualized with servers. In cloud some of the server may be available or not. It may be over loaded with user request. And sometimes remain offline or even crashed for various reasons. . Is this scenario, the users would look forward to the available server which is suitable to complete their request. Thus we should be able to offer an effective scheme to complete user requests. Most existing Failure Detector (FD) methods do not automatically adjust their detection service for dynamic network conditions. Fault tolerance is common phenomena in cloud computing network. So in order to determine the failures and satisfy user requests as well as to improve the Quality of Service (QoS) we propose a new scheme called self tuning failure detection (SFD) scheme in cloud environment. This proposed scheme works in cloud environment the corrects of which lies in inspecting the usr request before forwarding to available servers.

Keywords- Cloud computing; Virtualization; Failure detection; Fault-tolerance; Quality of Service

I. INTRODUCTION

Cloud computing is delivers the services through internet. Services in cloud allow users to use hardware and software managed at remote places. These services and computing resources are accessed from internet. Cloud provides set of resources, storage, and processing power, etc. In the cloud computing networks, example if subscriber want to store a file on cloud server at that point of time the server may be available or not. It may be over loaded with user request. And sometimes remain offline or even crashed for various reasons. We know that cloud service environment can be dynamic and unexpected, and users would expect the right and available servers to complete their application requirements. We must be able provide an effective scheme to handle service conditions and cloud resources. Thus fault-tolerant schemes are designed to provide reliable and continuous services in cloud computing networks despite the failures. As an essential building block for the cloud computing networks, a Failure Detector (FD) plays a critical role. Effective failure detection is essential to ensure QoS, and it is necessary to find an optimized FD that can detect failures in a timely and correct way before a generic FD service. The design of reliable FDs is a very difficult task. One of the main reasons is that the behavior of communication delays is unpredictable. The other reason is that asynchronous distributed systems make it impossible to determine precisely whether a remote process is failed or has just been very slow. An unreliable FD can make mistakes like falsely suspecting correct processes or trusting crashed processes. To ensure acceptable QoS for such an unreliable FD, parameters should be properly tuned to deliver a desirable QoS. Many fault-tolerant algorithms have been proposed, which are never-the-less based on unreliable FDs.

1.1 Cloud characteristics

- **On-demand:** A user can access the computing capabilities as needed. The capabilities are provided by cloud provider.
- **Broad network access:** The cloud facilities are accessed using different client platforms (thin/thick) and these facilities are available over the internet. The user can have the broad access to these facilities.
- **Resource pooling:** The computing resources are grouped together to serve multiple users at a time. These resources are dynamically assigned on user demand. The location of these servers is not known to consumer and has no control on it.
- **Measured service:** The services used by customer are automatically controlled by cloud systems. These systems are able to monitor, report the resource usage and provide the transparency to the provider and user.

1.2 Cloud delivery models

- **Software as a Service (Saas):** SaaS allow consumer to use the software which is already deployed over the internet. This software can be used through subscription. There is service level agreement between provider and consumer.
- **Platform as a Service (PaaS):** PaaS allows developing a new software or web applications easily. It avoids the complexity of buying and maintaining software.

- **Infrastructure as a Service (IaaS):** IaaS provides the computing infrastructure like server, storage, and operating systems as on user agreement. It avoids the purchasing of servers, software, storage space and network equipment instead of buying them.

1.3 Cloud deployment models

Cloud models represent the class of cloud environment and are divided based on size and access. It indicates the nature of the cloud.

- **Public Cloud:** In this model cloud services open for public user. Service provider gives the service and infrastructure to various users. The customers do not have any ability over the location of infrastructure. Compare to private cloud it is not secure.
- **Private Cloud**: It provides secure environment to user access. It permits only authorized users too control over the data. It is suitable for businesses which have unexpected needs, management demands and for those who need uptime requirements.
- **Hybrid Cloud**: It is combination of two or more cloud servers that it private or public or community cloud. They are combined together but remain identical entities. The benefits of this type of model are that deployed models are available in hybrid cloud. It increases the capability of cloud by aggregation with another cloud service.
- **Community Cloud**: This type of cloud belong particular community or an organization, for example bank. It can be shared with a number of organizations that belong to particular group which has similar work. The members of this cloud share privacy, performance and security among them.

1.4 Problem definition:

Failure Detection (FD) is an important role in cloud environment. An FD provides details about which servers are failed, even this information is not always correct because of message delay in dynamic network. Apart from this situation most of the methods cannot adjust their parameters according to dynamic network conditions. Some of these methods have to try all the possible parameters manually often if the network has major changes. Based on these parameters the QoS is decided, therefore it must satisfy the user needs. It is not easy task to fix the parameters for FDs in dynamic network, especially in cloud environment. Therefore we look for further technique for providing fault-tolerant for user request in cloud environment.

II. FAILURE IN CLOUD ENVIRONMEN

In cloud computing failure may occurs in several cases, for an example if a user want to store a file on cloud server, at that point of time the server may be available or not. It may be over loaded with user request. And sometimes remain offline or even crashed for various reasons. Is this scenario, as we already know that cloud environment is dynamic the users would look forward the available which is suitable to complete their request. Thus we should be able to offer an effective scheme to complete user requests.

2.1 Properties of failure detection:

- **Completeness:** When a process fails that process is eventually detected by at least one other non-faulty process. Completeness describes the capability of failure detector of suspecting every failed process permanently.
- **Accuracy:** Less number of faults results in high accuracy. It is impossible to build a failure detector that is 100% accurate and complete. Real life failure detectors guarantee 100% completeness but the accuracy is either partial or probabilistic.
- **Speed:** Time for the detection of failure should be as less as possible. In other words, time between occurrence of a failure and its prediction must be small.
- **Scale:** There should be low and equally distributed load on each process in overall network load.

III. RELATED WORK

To provide integration of software and resources which are dynamically scalable an adoptable technique is used [1]. In order to improve reliability and robustness in cloud computing. The faults are first detected and the suitable technique is applied to make system fault tolerant.

Increasing importance of high performance datacenters make them prone to attack and failures. These failures direct to thermal anomalies hotspots, which impact the cost of datacenters. The solution Thermal Anomaly-aware Resource Allocation (TARA) radically improves the detection probability compared to anomaly detection when predictable [3] scheduling algorithms are used. Sometimes there could be infrastructure change in cloud, these changes can lead to failures. It is shown that with sensitive rules for detecting failures, a use may be alerted to problems before provider make it announce [4].

The [5] wavelet-based multi-scale anomaly identification mechanism, Different from other anomaly identification approaches, it does not require a preceding knowledge of failure distributions, it can self-adapt by learning from observed failures at runtime that can evaluate cloud performance to identify anomalous cloud behaviors.

In dynamic network finding solution to user requests is not easy task, to avoid manual method the author proposed a new scheme self tuning failure detection [6], which can adjust the SFD control parameters to obtain corresponding service to satisfy user requirements, while maintaining good performance.

Another technique called Byzantine fault detection method for cloud computing is proposed, in which Petri nets are used to dynamically construct the components of cloud computing CFN(Cloud Computing Fault Net)[7]. It consists of system integration and analysis process. On this basis properties of model are analyzed, the operational semantics and theories of Petri Nets are help to prove the effectiveness of algorithm.

To overcome the drawbacks in large scale cloud network adaptive failure detection (AFD) is used. It does not require any prior failure history and it can self adapt by learning from observed failure events [8]. It is capable of finding failures not yet seen in the past. Based on the cloud performance data, AFD detects possible failures, which are verified by the cloud operators. They are confirmed as either true failures or normal states.

Failure occurrence becomes an important concern to system designers. The solution to this problem is the designer can make use of related data and troubleshoot it. It consumes much time, to overcome this new method autonomic mechanism for anomaly detection in cloud system [9] is proposed. The collected data are transformed to a standard format and the complexity of data is reduced by extracting the features.

In concerns to security issue how to reduce the impact of denial of Service (DoS) attack a new technique called a cooperative intrusion detection system is proposed [10]. It works by sending alert message to its neighbor system, these system could gather the same type of attack from other system, then it make judgment to resolve the trustworthiness of alert message. By cooperating among these, agents implements prevention technique.

3.1 Issues Identified:

- **Availability:** In the cloud computing networks, the servers should be available to users, but sometimes the servers may be active and available, while others are busy or heavy loaded and the remaining may be offline or even crashed for various reasons.
- **Fault-tolerant:** Fault tolerance provide reliable service to the user despite of failures of servers, it is one most challenging issue in cloud environment.
- **Scalable:** The services in cloud should be scalable means that the provider can able to provide as many numbers of servers.
- **Quality of Service (QoS):** The application should provide high level of QoS to user requirements in accessing the information remotely, and it should also ensure about the security.

IV. SCOPE OF THE PROJECT

As per as the research analyzed there is a potential need for implementing automatic failure detection by using different parameters in cloud environment to offer high Quality of Service (QoS) to the user requirement. To overcome these problems the new scheme is designed.

- The proposed scheme can handle dynamic network conditions.
- It can process the requirement of any number user requests that are running concurrently.
- The proposal can also provide the high availability of server at any point of time.
- Servers are categorized in two ways such as high and low configured servers, it is necessary because the Audio/Video(AV) file need high configured servers.
- If the server is overloaded then request is automatically switched to next available server.

V. SYSTEM MODEL AND IMPLEMENTATION

It is essential to maintain high level of QoS and should provide application security and dependability in accessing information from cloud servers. It is not an easy task to design reliable FDs; reason is that communication delays in dynamic network. Another reason in some cases we could not decide actually the server is crashed or been slow. In order to provide high QoS for unreliable FDs, it is necessary to set the parameters properly. To resolve this many algorithms are proposed, which are nevertheless based on unreliable FDs.

5.1 User classes and characteristics: The user and admin comes in the user classes. Only registered user can access the servers, and send the files to store in the servers at any time. The admin is responsible to inspect the request parameter and allocate the appropriate server to it, and can also create new servers as per user need. Admin can also view the registered user details and can also view the server details which are already allocated to requests.

5.2 Design and implementation constraints

- The system should install with NetBeans IDE and .NET frame work 4.
- In developed scheme each server is limited to 3 requests per server, Therefore user has to delete the uploaded files if it is not required.

5.3 Assumptions and dependencies

- User is not aware of servers which are going to allocate his requests.
- Admin is responsible for the selection of server to the request.
- Only registered user can send/download the files on to cloud.

5.4 Proposed Architecture

The goal of proposed work is to optimize the existing FDs, by designing SFD scheme that can manage dynamic network conditions and the any number of user requests that are running concurrently. In the proposed scheme the servers are categorize in two levels i.e. high and low configured server based on some threshold value (capacity and speed of server). Whenever the new request comes, it is inspects by considering predefined set of parameters and based on the result it looks forward for the suitable server. This is necessary because the AV files need high configured server to store and retrieve; else there could be unnecessary delay in processing. By doing this we can achieve better QoS. Basically there are 3 modules.

- User
- Administration
- SFD

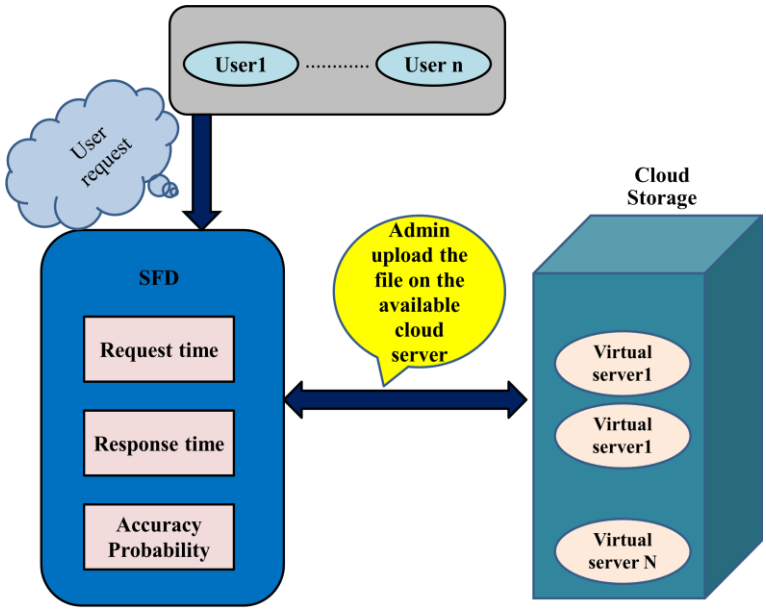


Figure 4.1 Architecture of proposed model

5.5 Implementation

Compared to general FD the method used in SFD is quite simple. When the user sends request to upload a file, the time counter is initiated, at end this starting time is used to calculate the total time taken to serve request. On the other side admin will check for arrived requests and forward those to available servers, and the timer is stopped, so that we can calculate the total time taken to serve the request. This total time should be less because it indicates the effectiveness of scheme. In this scheme each servers are able to store particular file formats, therefore it is SFP module responsibility to inspect the user request before forwarding to the available server. These servers are categorized into high or low capacity, by considering the server parameters such as size and speed of server, to decide this some threshold value is set. If the server capacity is more than the threshold value then it is considered as high else it is low capacity server. It is necessary to divide the servers because the video/audio file request needs high configured servers and low configured servers are used to store doc, pdf file request. If one of these servers has reached its maximum capacity then the new request is switched to next available server. User has facility to download the uploaded file or can delete if it is not required.

Functionality of each module

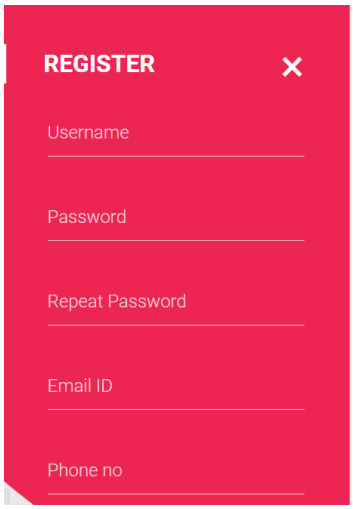
- **User Module:**
  - User registration: Allows new users to sign up by entering required information.
  - User login: The registered user can login with valid username and password.
  - Upload new file: After successful login user can able to upload a new file.
  - View uploaded files: User can view the files which are already uploaded on the servers.
  - Download/Delete the file: User can also have facility to download files, and can also delete if it is not required.

- **Administration or admin Module:**
  - View registered user details: Admin is capable to view the details of user, which are registered to this application.
  - Create virtual server: Admin has provided with the privilege that he can create new virtual servers as per the need.
  - View newly created server details: Admin can view details of servers.
  - Forward the user file on available server: Admin act as interface between user and the cloud servers, therefore it is admin responsible to forward the user request to available server.
  - View server details on which files are already uploaded: Admin has option to view details of server which are already allocated to user requests.

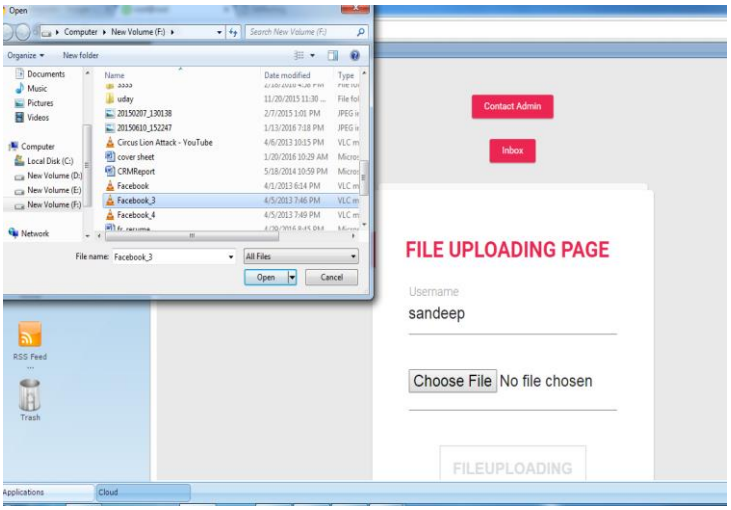
- **SFD Module:**
  - Inspect the user request: In this scheme each servers are able to store particular file formats, therefore it is SFD module responsibility to inspect the user request before forwarding to the available server.
  - Facilitates to choose the suitable server: The servers in cloud are categorized into high or low capacity, by considering the server parameters such as size and speed of server. And SFD helps admin to choose the suitable server for the request.
  - Calculate the total time: It is important increase the effectiveness of scheme, the effectiveness is measured with total time taken to serve request.

VI. RESULT ANALYSIS

Initially, all user and admin must be connected to the network. By connecting to the private cloud users can send the request to the admin So that he can serve the request and updates regularly. Before login the user must be register by entering all the required information as shown in snapshot 6.1. After the registration the user should login with valid username and password. Successful login directs the user to file upload page Figure 6.2. User can browse file from the system to upload, this request is sent to the admin. And the user is notified with message ‘file sent successfully’. User can also have the option to download/delete the files, which are already uploaded on the server.



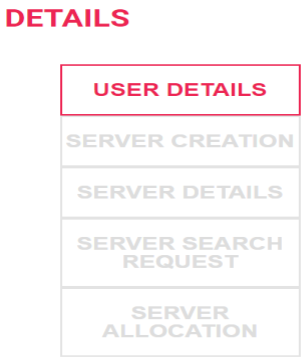
Snapshot 6.1User Registration



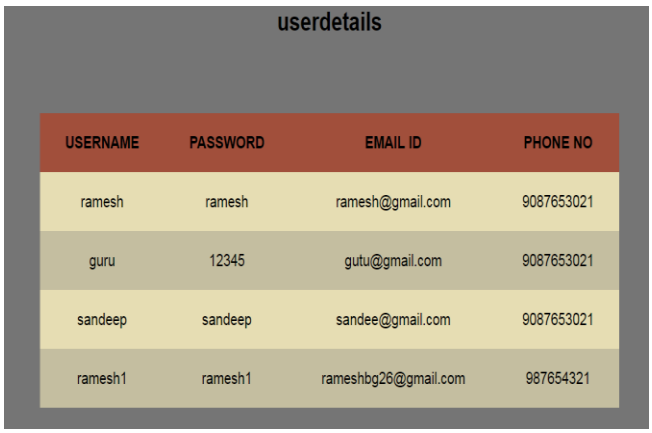
Snapshot 6.2File uploading

On the other side the admin should also login with valid Id and the password. After the successful login admin is directed to the admin privileges page (snapshot 6.3) where he can view the details of registered users (snapshot 6.4), create new virtual server and also view the details of servers.

The major part of proposed scheme is shown snapshot 6.5 here the admin can serve the user request so that it should be stored in the appropriate server. And the admin can also view the details of servers which are already served for request as shown in snapshot 6.6.

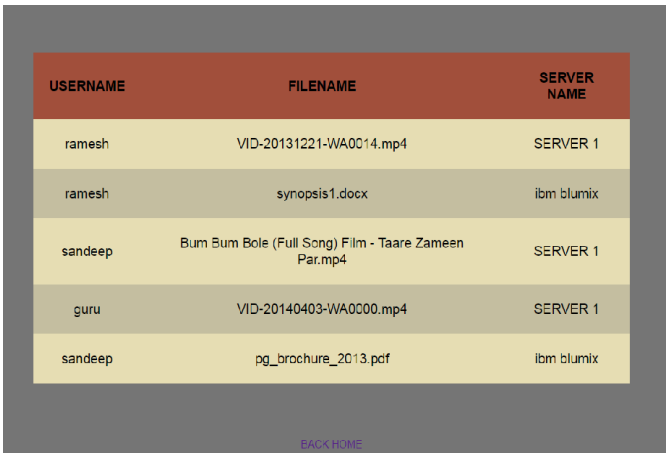


Snapshot 6.3 Admin privileges



Snapshot 6.4 User Details

Snapshot 6.5 forward file to the server



Snapshot 6.6allocated server details

CONCLUSION AND FUTURE ENHANCEMANT

In cloud service, servers are virtualized with network. The servers may be active or not, they may be over loaded with user request. And sometimes remain offline or even crashed for various reasons. And most of existing scheme do not adjust to dynamic network conditions, therefore they could not be used for real time applications. In this scenario, it would prompt the user to look forward available server. In order to provide automatic fault tolerance and non manual analysis to user requirement the new scheme self tuning failure detector is proposed. Experimental results demonstrate that this method can work in dynamic network condition to attain subsequent services and satisfy user requirements, whereas maintaining good performance. It can be used in businesses and commercial purpose as a part of future enhancement because it is beneficial to cloud networks.

REFERENCES

[1] Zeeshan Amin, Nisha Sethi, Harshpreet Singh, " Review on Fault Tolerance Techniques in Cloud Computing", International Journal of Computer Applications (0975 – 8887) Volume 116 – No, 7, 18, April 2015

[2] Jing Liu, Jiantao Zhou, Rajkumar Buyya, "Software Rejuvenation based Fault Tolerance Scheme for Cloud Applications", IEEE 8th International Conference on Cloud Computing, 4, 2015

[3] Eun Kyung Lee, Hariharasudhan Viswanathan, Dario Pompili, "Model-based Thermal Anomaly Detection in Cloud Datacenters using Thermal Imaging", 2481423 IEEE, 15, 2015

[4] Rabia Chaudry, Adnene Guabtni, Alan Fekete1, Len Bass, Anna Liu, "Consumer Monitoring of Infrastructure Performance in a Public Cloud", Springer International Publishing Switzerland Part II, LNCS 8787, pp. 423–432, 10, 2014

[5] Qiang Guan and Song Fu, Nathan DeBardeleben and Sean Blanchard, "Exploring Time and Frequency Domains for Accurate and Automated Anomaly Detection in Cloud Computing Systems", IEEE DOI 10.1109/PRDC, 10, 2013

[6] Naixue Xiong, Athanasios V. Vasilakos, Jie Wu, Y. Richard Yang, Andy Rindos, Yuezhi Zhou1,Wen-Zhan Song2,Yi Pan2, "A Self-tuning Failure Detection Scheme for Cloud Computing Service", IEEE DOI 10.1109/IPDPS, 12, 2012

[7] Guisheng Fan, Huiqun Yu, Liqiong Chen, Dongmei Liu, "Model Based Byzantine Fault Detection Technique for Cloud Computing", IEEE DOI 10.1109/APSCC, 8, 2012

[8] Husanbir S. Pannu and Jianguo Liu, Qiang Guan and Song Fu, "AFD: Adaptive Failure Detection System for Cloud Computing Infrastructures", IEEE 978-1-4673-4883-6, 10, 2012

[9] Derek Smith, Qiang Guan, and Song Fu, "An Anomaly Detection Framework for Autonomic Management of Compute Cloud Systems", IEEE DOI 10.1109/COMPSACW, 6, 2010

[10] Chi-Chun Lo, Chun-Chieh Huang, Joy Ku, "A Cooperative Intrusion Detection System Framework for Cloud Computing Networks", IEEE DOI 10.1109/ICPPW, 5, 2010