



ISSN: 2230-9926

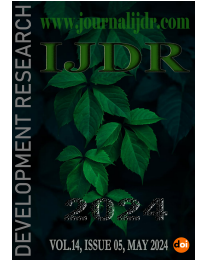
Available online at <http://www.journalijdr.com>

IJDR

International Journal of Development Research

Vol. 14, Issue, 05, pp. 65755-65760, May, 2024

<https://doi.org/10.37118/ijdr.28346.05.2024>



RESEARCH ARTICLE

OPEN ACCESS

ADAPTIVE VIRTUAL INSTANCE CONSOLIDATION FOR ENHANCED LOAD BALANCING IN CLOUD DATA CENTERS

*Balaji, C.

Assistant Professor, Department of Computer Science, SRM Institute of Science & Technology, Trichy Campus

ARTICLE INFO

Article History:

Received 11th February, 2024

Received in revised form

26th March, 2024

Accepted 09th April, 2024

Published online 30th May, 2024

Key Words:

Cloud computing, virtualization, green computing, workload requirements, server utilization, Virtual Machine (VM) technology, resource allocation, job response times, task execution, VM live migration, cloud resources, workload balancing, system slowdown prevention, VM consolidation technique, co-location strategy, self-destruction strategy, real-time resource allocation, online prediction model, partition sizes, reducer allocation.

*Corresponding author: Balaji, C.

ABSTRACT

Cloud computing offers significant benefits to commercial clients by efficiently handling expanding workloads in a planned manner. The primary enabling technology for cloud computing is virtualization, which simplifies infrastructure management and usage. This paper leverages virtualization to promote green computing and allocate resources based on workload requirements. By minimizing skewness and mixing diverse workloads, server utilization is increased. However, managing client demand poses challenges, which are addressed through Virtual Machine (VM) technology to dynamically allocate resources. The implementation of a virtualized environment is expected to reduce job response times and optimize task execution based on resource availability. VMs are assigned to users according to their needs, and VM live migration facilitates VM and Physical Machine (PM) mapping. Efficient utilization of cloud resources helps balance workloads and prevent system slowdowns. A local negotiation-based VM consolidation technique is employed to anticipate task requests and generate virtual space. A co-location strategy combines small, empty rooms to enhance server performance, while a self-destruction strategy eliminates inaccurate data based on time-to-live property. Real-time resource allocation and an online prediction model aid in determining partition sizes for reduction jobs and dynamically allocating resources to reducers with large partitions to expedite task completion.

Copyright©2024, Balaji et al., This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Citation: Balaji, 2024. "Adaptive Virtual Instance Consolidation for Enhanced load Balancing in Cloud Data Centers". International Journal of Development Research, 14, (05), 65755-65760.

INTRODUCTION

Cloud storage is a revolutionary model of data storage that offers a flexible and efficient solution for storing digital data. In this innovative approach, data is stored in logical pools across multiple servers, often in different locations, all managed by a hosting company. These cloud storage providers ensure that data is always available, secure, and easily accessible. Individuals and organizations can purchase or lease storage capacity from these providers to store various types of data, whether it be user information, organizational data, or application data. Cloud storage services can be accessed through various means, such as cloud computer services, web service APIs, or applications that utilize these APIs. This highly virtualized infrastructure offers near-instant elasticity, scalability, multi-tenancy, and metered resources, making it a popular choice for many users. Whether utilizing off-premises services like Amazon S3 or on-premises solutions like ViON Capacity Services, cloud storage provides a versatile and reliable option for data storage needs. Object storage services, software, and systems, as well as distributed storage research projects, all exemplify the diverse range of cloud storage options available today.

Reason for the Project: This paper addresses a significant challenge of efficiently allocating resources based on workload demands, which is crucial for commercial clients relying on cloud services for their computing requirements. Through the use of Virtual Machine (VM) technology, the project aims to offer resources as needed, ensuring clients have the necessary computing power available when required. Moreover, the project introduces the concept of "skewness" to minimize workload imbalances by combining various workloads to enhance server utilization and optimize resource distribution. This is especially vital in cloud settings where resources are shared among multiple users and applications. Overall, the project's emphasis on virtualization, green computing, and dynamic resource allocation seeks to enhance cloud computing efficiency, lower costs for commercial clients, and boost overall system performance.

Problem Statement: One of the most considerable challenges in the cloud computing realm is the need for efficient resource management to support the dynamic and diverse needs of business clients. Even though numerous enterprises favor cloud environments because they provide scalability and flexibility, their use is hardly sustainable. In particular, workloads are flexible, and the need is continuously changing is the primary issue, which logically leads to the

consumption of massive resources and costs. Resource allocation frequently consumes suboptimal possibilities under the conditions and remains the primary source of consumption and costs. Lack of adaptive approaches to cope with changing workloads and client needs is a fundamental hindrance to cloud services in general performance and scalability especially resource allocation. To address the problem, it is proposed to consider new approaches based on virtualization technology to advance existing resource management practices, support the principles of green computing, and improve resource allocation markets.

Aim & Objective: The main goal of this paper is to transform how resources are allocated and used efficiently in cloud computing, with a focus on meeting the intricate needs of business customers. Through leveraging virtualization technology and adopting eco-friendly computing practices, the project aims to improve resource usage, reduce expenses, and boost performance in cloud settings. Implement cutting-edge virtualization solutions to abstract physical resources, creating a flexible and manageable computing environment. Advocate for green computing by formulating resource allocation strategies that optimize energy efficiency and promote sustainable practices in cloud computing. Develop dynamic resource allocation algorithms to align resources with fluctuating workloads and evolving client requirements. Devise cost optimization strategies to minimize costs for commercial clients while ensuring high performance and reliability. Enhance client satisfaction by providing reliable, scalable, cost-effective, and customized cloud services tailored to each client's specific needs.

Scope: The aim of this paper is to improve resource allocation and efficiency in cloud computing, with a focus on meeting the requirements of commercial clients. Virtualization technology will be utilized to establish a more adaptable and controllable computing environment, allowing for dynamic resource allocation based on workload demands. The project will advocate for green computing practices to enhance energy efficiency and minimize environmental impact. Strategies for cost optimization will be devised to reduce expenses for both cloud service providers and clients. Additionally, the project will strive to enhance client satisfaction by delivering dependable, scalable, and cost-efficient cloud services tailored to their specific needs. An evaluation of performance will be carried out to gauge the effectiveness of the proposed solutions, and recommendations will be offered to cloud service providers to refine their resource management practices and boost service efficiency.

System Tools: PHP, which stands for Hypertext Preprocessor, is a popular scripting language used for various purposes, originally intended for creating dynamic web pages. In web development, PHP code is integrated into HTML documents and interpreted by a web server equipped with a PHP processor module to generate the final web page. Besides web development, PHP can also be used as a general-purpose programming language, running through an interpreter application in command-line mode to perform system operations and produce program output. It can even function as a graphical application. PHP is widely supported, available as a processor for modern web servers and as a standalone interpreter across various operating systems and platforms. Developed by Rasmus Lerdorf in 1995, PHP has undergone continuous evolution, with The PHP Group now leading its main implementation, serving as the unofficial standard due to the absence of a formal specification. PHP is open-source software distributed under the PHP License, distinct from the GNU General Public License due to specific usage restrictions related to the term PHP. The term "Hypertext" denotes interconnected files using hyperlinks, like HTML files, while "Preprocessing" involves executing instructions that alter the output. Below, we illustrate the distinctions between HTML and PHP files.

Accessing a PHP Page

- Your browser sends a request to that web page's server for the PHP file you wish to view.
- The web server calls PHP to interpret and perform the operations

called for in the PHPscript.

- The web server sends the output of the PHP program back to your computer.
- Your browser displays the output appropriately.

Benefit of PHP: Due to the server's processing capabilities, the output of PHP files is subject to change based on alterations in their input. For instance, the majority of pages within the Horticulture website feature a pair of PHP directives:

1. The inclusion of the header file, which defines the left-hand links, the banner, and the top quick links.
2. The inclusion of the footer file, which showcases the mission statement and contact details for Horticulture.

As the files are included each time the PHP file is accessed, any modifications to the header/footer files will result in an immediate update of the content. Essentially, the addition of a new link will be promptly reflected on all pages that incorporate the header.

Security: Approximately 30% of vulnerabilities listed on the National Vulnerability Database are associated with PHP. These vulnerabilities often arise from not adhering to best programming practices rather than inherent flaws in the language or its core libraries, which are relatively rare (only 23 in 2008, accounting for about 1% of the total). To address the inevitability of human error, some programming languages incorporate taint checking to automatically identify input validation shortcomings that lead to numerous issues. While efforts to introduce this feature to PHP have been repeatedly denied in the past, there are alternative protective measures like Suhosin and Hardening-Patch tailored for securing Web hosting environments. PHPIDS enhances the security of PHP applications by safeguarding against various intrusions, including cross-site scripting (XSS), SQL injection, header injection, directory traversal, remote file execution, remote file inclusion, and denial-of-service (DoS) attacks.

Data types: PHP stores whole numbers in a platform-dependent range, either a 64-bit or 32-bit signed integer equivalent to the C-language long type. Unsigned integers may be converted to signed values in specific cases, a behavior that sets PHP apart from other programming languages. Integer variables can be assigned using decimal (positive and negative), octal, and hexadecimal notations. Floating-point numbers are also stored within a platform-specific range, specified using floating-point notation or two forms of scientific notation. PHP features a native Boolean type akin to those in Java and C++, following conversion rules where non-zero values are considered true and zero as false, similar to Perl and C++. The null data type signifies a variable without a value, with the only valid value being NULL. Variables of the "resource" type denote references to external resources, generated by extension-specific functions and exclusively handled by functions within the same extension, such as file, image, and database resources. Arrays in PHP can encompass elements of any type the language supports, including resources, objects, and even other arrays. Lists of values and hashes with keys and values maintain order, allowing for a mix of both types within arrays.

Inter images: MySQL is mainly known as a relational database management system (RDBMS) and does not come with graphical user interface (GUI) tools for managing databases or data. Users can utilize command line tools provided or opt for MySQL "front-ends" such as desktop software and web applications to handle tasks like creating and managing databases, structuring databases, backing up data, checking status, and working with data records. One popular front-end tool, MySQL Workbench, is continuously developed by Oracle and is accessible for free.

Graphical: The official MySQL Workbench, developed by MySQL AB, is a free integrated environment that allows users to visually administer MySQL databases and design database structures. It has replaced the previous software package, MySQL GUI Tools. MySQL

Workbench is considered the authoritative MySQL frontend, offering features for managing database design, modeling, SQL development (replacing MySQL Query Browser), and database administration (replacing MySQL Administrator). There are two editions available: the free and open-source Community Edition, which can be downloaded from the MySQL website, and the proprietary Standard Edition that enhances the features of the Community Edition.

Feasibility Study: A feasibility study assesses the practicality and endurance of a concept, project, or business. It evaluates the availability of resources for execution and the potential for generating satisfactory profits. Moreover, it showcases the advantages gained by undertaking the risk of investing in the idea. These assessments scrutinize the strengths, weaknesses, opportunities, and threats to ascertain the cost-effectiveness and advantages of the proposals for a company's sustained growth. Additionally, investors can gain insights by reviewing the challenges and remedies outlined in the study to decide if a proposed project aligns with a company's objectives.

Technical Feasibility: This evaluation is centered around an initial system requirements design to ascertain the company's capability in handling the project. When drafting a feasibility report, it is essential to consider the following aspects:

- A concise overview of the business to analyze various factors that may impact the study
- The specific area of the business under scrutiny
- The human and economic elements involved
- Potential solutions to address the issue

The primary focus at this stage is to determine the technical and legal feasibility of the proposal, assuming a reasonable cost. The technical feasibility evaluation aims to assess the organization's existing technical resources and their suitability for the proposed system's requirements. This assessment involves examining the hardware and software to ensure they align with the proposed system's needs.

Operational Feasibility Study: Operational feasibility evaluates how effectively a proposed system addresses issues, capitalizes on opportunities outlined during scope definition, and meets the requirements identified in the system development's requirements analysis phase. The assessment of operational feasibility centers on determining how well the proposed project aligns with the current business environment and goals concerning development timeline, delivery date, corporate culture, and existing business procedures.

Economic Feasibility: Assessing the economics of an investment is a crucial aspect of investment analysis, focusing on factors that can be quantified, measured, and compared in monetary terms (Chen 1996). The outcomes of this evaluation are weighed alongside other considerations to determine the viability of the investment project. A thorough investment appraisal ensures that the most promising projects are pursued, increasing the likelihood of success. Investment projects involve allocating capital and resources to generate future returns, whether through profits, cost reductions, or societal gains. To be deemed worthwhile, the anticipated benefits should exceed the resources invested to attain them.

Existing System: In this study, we explore the concept of stochastic load balancing through VM migration to tackle demand uncertainty and fluctuating workloads. Unlike previous research, our approach involves probabilistically defining the resource demands of VMs and the load statuses of PMs. The primary goal is to prevent resource overutilization on each PM, ensuring that each resource type remains within its capacity with a high level of certainty, as dictated by the SLA agreement. This probability reflects the potential risk of SLA breaches on individual PMs. By employing stochastic workload characterization, we can effectively address the variability and changes in resource usage. The implementation of probabilistic assurances for managing overloads allows for more resilient application performance in the face of dynamic workloads, while also optimizing resource utilization through statistical multiplexing.

However, this stochastic load balancing approach presents new challenges, such as accurately estimating stochastic resource demands, identifying hotspots, and executing VM migrations that account for multidimensional stochastic resource requirements.

Drawbacks of Existing System

- Workloads are not handle properly
- Loads are queued in storage system so provide low progress rate.
- Overhead occurred at the time of repartitioning the data

Proposed System

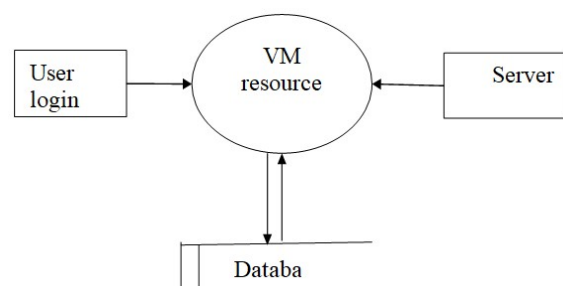
This paper introduces a framework called the VM consolidation mechanism to dynamically allocate resources. Despite this advancement, current schedulers face an issue known as partitioning skew, where map task outputs are unevenly distributed across systems. The VM consolidation mechanism in this paper addresses partitioning skew in real-time by adjusting task resource allocation, eliminating the need for data repartitioning. Two key challenges in this mechanism include developing a runtime forecasting algorithm to predict reducer partition sizes and creating a task performance model to determine the appropriate container size for each reduce task. Our approach simplifies partition size prediction, enabling high-quality results without modifying the partitioning implementation. Additionally, we propose implementing Co-Located VMs to optimize resource utilization and introduce a self-destruction approach in Cloud Storage to manage data expiration efficiently. This system integrates active storage techniques to enhance data management.

Benefits of Proposed System

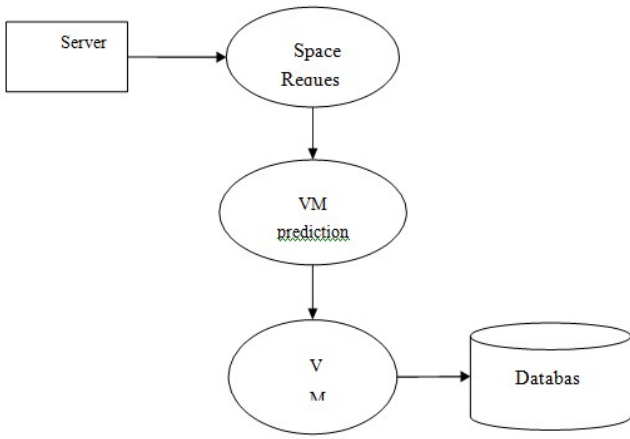
- Balanced load predicted approach
- Overcome the degradation of server performance
- Efficient load balancing at the time of overflow the jobs

Data Flow Diagram: A data flow diagram illustrates how information moves within a process or system, showcasing data inputs and outputs, data storage, and the different stages the data passes through. These diagrams utilize standardized symbols and notation to depict entities and their connections. They provide a visual representation of complex systems and processes that may be challenging to explain solely through text. By utilizing DFDs, you can analyze and enhance existing systems or design new systems effectively. The visual depiction of each component aids in pinpointing inefficiencies and creating optimal systems.

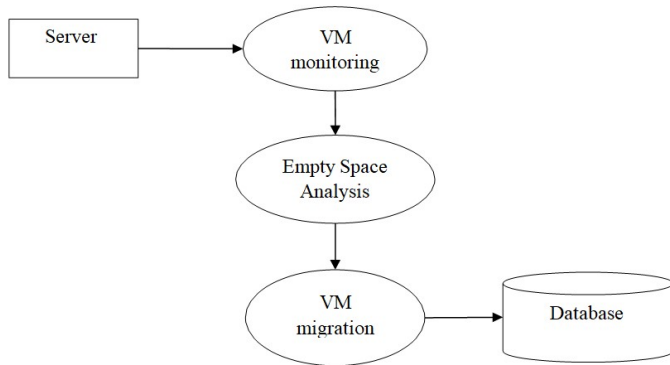
Level 0: This diagram, often referred to as a context diagram, serves as an abstraction view that illustrates the system as a unified process and its connections to external entities. It depicts the entire system as a single bubble, with incoming and outgoing arrows indicating input and output data.



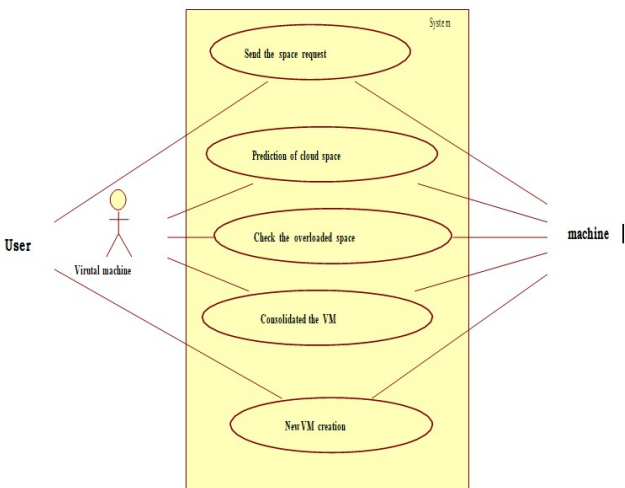
Level 1: In a 1-level DFD, the context diagram is divided into several bubbles or processes. At this stage, we focus on the primary functions of the system and break down the overall process from the 0-level DFD into smaller sub processes.



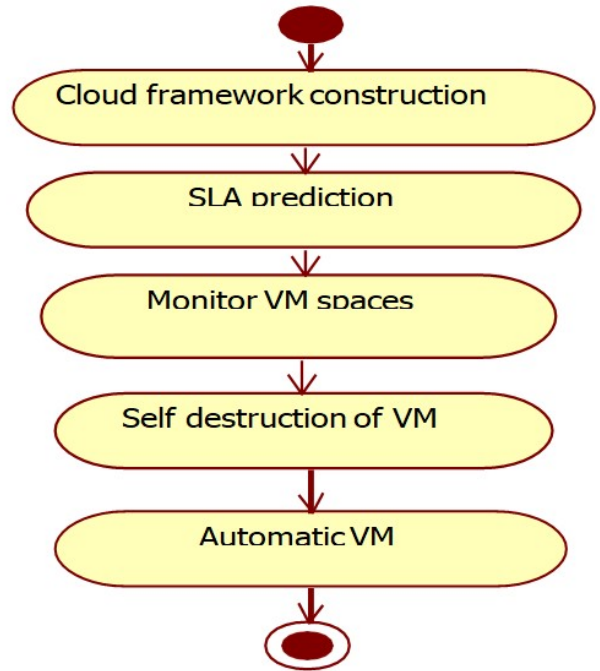
Level 2: A 2-level DFD delves further into the components of a 1-level DFD, providing a more detailed view of the system's operations. This deeper level of detail is useful for planning and documenting specific aspects of the system's functionality.



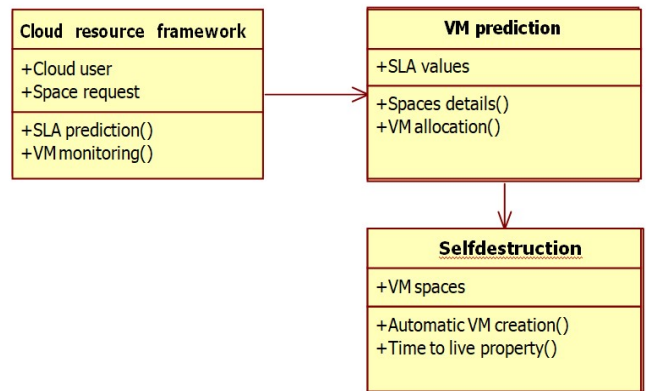
Usecase Diagram: A use case diagram, at its core, shows how a user interacts with a system, highlighting the relationships between the user and various use cases. Here, the term "system" pertains to something under development or operation, such as a website. The "actors" represent individuals or groups playing specific roles within the system.



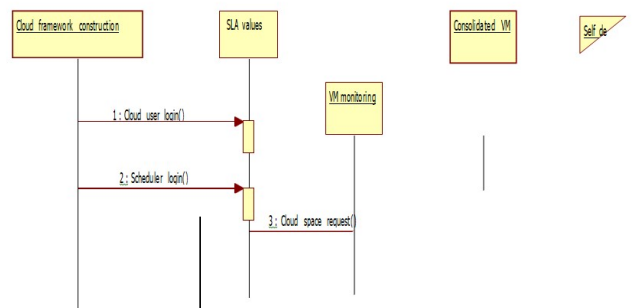
Activity Diagram: The activity diagram displays a distinct type of state diagram where most states are action states and transitions are mainly triggered by actions being completed in the source states. These actions can be referred to as system operations, leading to the flow of control moving between operations. This flow can happen concurrently, in parallel, or sequentially. Activity diagrams incorporate different elements to handle different types of flow control.



Class Diagram: A class diagram, according to the Unified Modeling Language (UML), is a static structural diagram that showcases a system's classes, properties, functions, and object interactions. The class diagram is a key element in object-oriented modeling, serving purposes in both technical modeling, where it aids in translating models into computer code, and in conceptual modeling for systematically designing applications.



Sequence Diagram: An object's interactions are arranged chronologically in a sequence diagram. It shows the classes and objects that are a part of the scenario as well as the messages that are passed between the objects in order for the scenario to work. Sequence diagrams are commonly linked to the realizations of use cases in the Logical View of the system that is being developed. Event diagrams or event scenarios are other names for sequence diagrams



Module Description

- Cloud resource framework
- SLA prediction
- VM monitoring
- Co-located VM approach
- Self-destruction
- Encrypted Data Storage

Cloud Resource Framework: Cloud computing is seen as a novel computing paradigm that is designed to offer dependable, tailored, and QoS (Quality of Service) assured computing environments that are dynamic for end-users. The emergence of cloud computing is a result of the convergence of distributed processing, parallel processing, and grid computing. At the core of cloud computing lies the fundamental principle that user data is not stored locally but rather in the internet data center. The operation and maintenance of these data centers are handled by companies that offer cloud computing services. Users have the ability to access their stored data at any given time through any internet-connected terminal device by utilizing the Application Programming Interface (API) provided by cloud service providers. Within this framework, the cloud system is initialized by multiple entities including cloud users, physical machines, and virtual machines. The allocation of resources within the cloud system is the responsibility of the physical machine.

SLA Prediction: In a research report, a service-level agreement (SLA) represents a formal agreement between a physical machine and its users, outlining the resources that the physical machine will provide and setting performance standards. The allocation of resources to the physical machine is achieved by inputting the SLA values. SLAs play a crucial role in managing customer expectations regarding the performance and quality of service provided by the service provider. These agreements may define various metrics to measure performance, such as:

VM Monitoring: Within this module, there are two sub-modules: VM prediction and Migration plan. The VM prediction sub-module is utilized to ascertain the specifics regarding allocated and available storage space within storage environments. Conversely, the Migration plan sub-module is employed to transfer control from one VM to another. The identification of Virtual and physical machines is consistently determined by space thresholds established either by the data center proprietor or in accordance with the Service Level Agreements outlined by clients. Physical machines exceeding the upper threshold for resource utilization are identified as hotspots, while those falling below the lower threshold are categorized as servers. The former indicates over-utilization, whereas the latter signifies underutilization, a concept applicable to various resource dimensions.

Co-Located vm approach: In this module, the analysis of the pending small spaces within each virtual machine is conducted. The user's Service Level Agreement (SLA) is verified against the available spaces. In cases where the existing spaces are insufficient, the pending spaces are aggregated to form new virtual machine space. Subsequently, resources are allocated in new spaces by the physical machine based on this consolidation. Virtual Machines (VMs) are defined as individual instances of an operating system running along with one or more applications within an isolated partition of the computer. Multiple virtual machines can operate on a single physical machine simultaneously. In situations where a physical host becomes overloaded, there may be a need to transfer a portion of its load dynamically to another machine with minimal disruption to users. This process of relocating a virtual machine from one physical host to another is known as migration. Historically, the relocation of a VM between two physical hosts necessitated the shutdown of the VM, allocation of required resources to the new physical host, transfer of VM files, and initiation of the VM on the new host.

Self-Destruction: This module, the time-to-live variable is calculated to forecast the validity of each user. If the validity date expires, an alert is sent one day prior to prompt the extension of resource space.

Resource space is allocated in the VM machine if the user extends resources; otherwise, all allocated resources in the VM for users are eliminated. A self-destruct method is employed to remove data from the cloud storage in accordance with the defined rules. Upon specifying the survival time, users' data will be deleted from the cloud environment once the survival time elapses. The self-destruct operation is triggered by a set of predefined rules. The user object possesses an indefinite lifespan, meaning it will persist until manually deleted by the user. The time-to-live value of an active storage object is constrained, resulting in the deletion of an active object when the associated Policy object's value becomes true.

Encrypted data Storage: Data owner could upload the file on cloud. Once the file is stored in cloud, the file will get encrypted. Encryption is a popular technique that plays a major role to protect data from intruders. AES algorithm uses a particular structure to encrypt data to provide the best security. To do that it relies on a number of rounds and inside each round comprise of four sub-process. The AES encryption algorithm defines a number of transformations that are to be performed on data stored in an array. The first step of the cipher is to put the data into an array; after which the cipher transformations are repeated over a number of encryption rounds. The number of rounds is determined by the key length, with 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys.

Implementation: Testing software in a "black box" means that the tester is not aware of the inner workings, architecture, or language of the module under test. Black box tests, like the majority of other test types, need to be created from a definite source document, such a requirements or specification document. In this type of testing, the software being tested is handled like a black box. One is unable to "see" into it. Without taking the software's operation into account, the test generates inputs and reacts to outputs, realization. Languages used for programming allow humans and machines to communicate. Features of programming languages and coding styles have a significant impact on the maintainability and quality of software. The following features are taken into consideration during coding.

- Translation from design to code is simple
- Memory efficiency
- Code efficiency
- Upgradability

The project's implementation phase is when the theoretical design is transformed into a functional system. As a result, it might be said to be the most important phase in creating a new system that is successful and instilling trust in the user regarding its functionality.

Maintenance: System maintenance is a critical aspect of managing any technological infrastructure. It encompasses a range of activities that are essential for the smooth and efficient operation of a system. One key aspect of maintenance is monitoring system performance. This involves tracking metrics such as CPU usage, memory utilization, and disk space to identify any potential issues or bottlenecks. By regularly monitoring performance, system administrators can proactively address issues before they escalate and impact the system's availability or performance. Another important aspect of system maintenance is applying software updates and patches. Software vendors regularly release updates to address security vulnerabilities, improve performance, and add new features. It is crucial to keep the system up to date with these patches to ensure that it remains secure and stable. Additionally, performing regular backups of data is essential for protecting against data loss due to hardware failure, human error, or cyber attacks. Backups should be stored securely and tested regularly to ensure that they can be successfully restored if needed. System maintenance also involves conducting regular security audits and vulnerability assessments to identify and address potential security risks. This includes implementing security best practices such as using strong passwords, enabling firewalls, and regularly updating antivirus software. By maintaining a strong security posture, organizations can protect their systems and data from unauthorized access and cyber threats.

Testing Definition: Software testing is conducted on a fully integrated system to assess the system's compliance with the corresponding requirements. In system testing, components that have passed integration testing are utilized as input. The objective of integration testing is to identify any discrepancies between the integrated units. Defects within both the integrated units and the entire system are identified through system testing. The outcome of system testing is the observed behavior of a component or system during testing. System Testing is executed on the entire system within the framework of either system requirement specifications, functional requirement specifications, or both. System testing evaluates the design, behavior, and customer expectations of the system. It is carried out to assess the system beyond the limits specified in the software requirements specification (SRS). System Testing is primarily conducted by an independent testing team separate from the development team to ensure impartial evaluation of the system's quality. This testing encompasses both functional and non-functional aspects and is classified as black-box testing. System Testing takes place after integration testing and before acceptance testing.

Testing Objective: Testing, an activity systematically planned and conducted, is initiated at the module level and progresses towards the integration of the entire computer-based system. The system's success is contingent upon testing, as it constitutes an essential component.

Several principles can function as testing objectives

- The process of executing a program with the aim of identifying errors is testing.
- An ideal test case is characterized by a high likelihood of discovering an unidentified error.
- A test is deemed successful when it reveals an unidentified error.

If testing is carried out successfully in accordance with the aforementioned objectives, it will expose errors in the software. Additionally, testing validates that the software functions align with the specifications and that performance requirements have been satisfied.

There are three methods for testing a program:

- For accuracy
- For implementation efficiency
- For computational complexity.

Experiments aimed at assessing implementation efficiency seek to identify methods for enhancing the speed or reducing the storage requirements of a program. This process involves refining the code and reevaluating the implementation phase within algorithm development. Computational complexity tests involve conducting an empirical analysis of an algorithm's complexity or comparing multiple algorithms that address the same issue. Data is inputted in various formats individually, with immediate correction of any errors detected. A quality team appointed by the management validates essential documents and conducts software testing across all levels during data entry.

CONCLUSION

In the realm of Cloud Computing, a Resource Allocation System (RAS) can be perceived as a mechanism designed to ensure that the requirements of applications are appropriately addressed by the infrastructure of the provider.

In addition to providing this assurance to developers, mechanisms for resource allocation should also take into account the present state of each resource within the Cloud environment. This consideration enables the application of algorithms that enhance the allocation of physical and/or virtual resources to developers' applications, thereby reducing the operational expenses of the cloud environment. Our system dynamically multiplexes virtual resources to physical resources in response to fluctuating demands. The Migration method is employed to effectively merge virtual machines (VMs) with varying resource attributes, optimizing the utilization of server capacities. The proposed algorithm not only prevents system overload but also promotes green computing practices in systems with multiple resource constraints.

Future Enhancements: In future the work can be extended to implement the proposed approach in a real time cloud environment by considering economy based preemption policies. And implement various VM provisioning scheduling approaches to improve the energy consumptions.

REFERENCES

- Gulshan Soni and Mala Kalra, "A Novel Approach for Load Balancing in Cloud Data Center", IEEE 2014.
- K. Balaji, Sai Kiran.P and Sunil Kumar.M,"Load Balancing in Cloud Computing: Issues and Challenges, Turkish Journal of Computer and Mathematical Education. Vol.12 No2 (2021), 2077 – 3084
- M. Hines and K. Gopalan, "Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning", in Proceedings of the ACM/Usenix International Conference on Virtual Execution Environments (VEE'09), pp.51-60, March 2009.
- M. Nelson, B. Lim, and G. Hutchins, "Fast transparent migration for virtual machines", in Proceedings of the annual conference on USENIX Annual Technical Conference , pp. 25-25, April 2005.
- M. Randles, D. Lamb, and A. Bendiab, "A comparative Study into distributed load balancing algorithms for cloud computing", IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp.551-556, April 2010.
- S. Sharma, S. Singh, and M. Sharma, "Performance analysis of Load Balancing Algorithms", World Academy of Science, Engineering and Technology (WASET), vol. 14, pp. 248-251, February 2008.
- Soumya Ranjan Jena, Sudarshan Padhy and Balendra Kumar Garg, "Performance Evaluation of Load Balancing Algorithms on Cloud Data Centers.
- W. Bhatiya, "CloudAnalyst a CloudSim-based tool for modelling and analysis of large scale cloud computing environments", MEDC Project, Cloud Computing and Distributed Systems Laboratory, University of Melbourne, Australia, pp. 1-44, June 2009.
- Y. Teo and R. Ayani, "Comparison of load balancing strategies on cluster-based web servers, Simulation", the journal of the Society for Modeling and Simulation, vol. 77, pp. 185-189, November-December 2001.
- Z. Zhang and Xu. Zhang "A Load balancing mechanism based on ant colony and complex network theory in open cloud computing federation", 2nd International Conference on Industrial Mechatronics and Automation (ICIMA), Wuhan, China, vol. 2, pp.240-243, May 2010.
