International Journal of Computer Applications (0975 – 8887)
Volume 184 – No.12, May 2022

Resource Allocation in Cloud: History Kerberos based Approach

Saleem Basha
Assistant Professor
Computing and Informatics Department
Mazoon College

Mohamed Nasar
Assistant Professor
Computing and Informatics Department
Mazoon College

ABSTRACT
Cloud computing is one of the remarkable approaches for massive computation. The cloud has solved many computational issued faced in the past. It has brought more benefits for the individuals those cannot offer the necessary computation power for their work and research. In this aspect, several cloud-based organizations have bloomed to server for the needs. As the demand of the cloud computation increases it is necessary to cope with the resources and efficient resource allocations. It is practically to have enormous resources to serve each and individual request. Instead, the best optimal solution is to have minimal resources with optimal resource scheduling algorithms. This research article proposed a history Kerberos based resource allocation techniques.

Keywords
History Kerberos, Cloud, Resources

1. INTRODUCTION
Resource allocation [2] is a very important topic that has been discussed in many areas of IT, such as software engineering, operating systems, grid computing, data management centers (data center) and cloud computing. The most important goals of resource allocation are to integrate unused resources to create shared and virtualized resource pools, make access to user demand to resources conveniently, improve resource utilization, etc. However, the available supplier resources and consumer resource requirements are both dynamically varied. Therefore, defining a mechanism for allocating resources to users in a flexible, dynamic and reliable way is one of the main challenging tasks in many areas of IT such as Cloud Computing[29].

Nowadays, there has been a dramatic increase in the popularity of cloud systems computing that offer computing resources on demand, based on billing a usage, so that users can increase or decrease their consumption rate resources according to their needs. These environments can multiplex many users on the same physical infrastructure [1],[26]

In cloud computing environments, there are two important stake holders: providers and cloud computing users [27][30]. On the one hand, suppliers hold resources like massive computing in their large data centers and rent these resources to the users. On the other hand, there are users who have applications with varied loads and take resources from vendors to run their applications. First of all, the user sends a request contains the resource requirements to the provider. After receiving the request, it looks for resources to satisfy the request and allocates these resources to the requesting user, usually in the form of virtual machines (VM). Then the user uses the resources assigned to him to run their applications and pays the fees of resources that used. When the user ends up with these resources, they returned to the provider. One of the interesting aspects of cloud computing is that these players have their own interests. As a general rule, the objective of suppliers is to maximize their income with a minimum investment. To this end, suppliers want to maximize the use of their IT resources, for example, by hosting the most as many virtual machines as possible on each machine[31]. On the other hand, the users want to get their work done at minimal cost or, in other words, they want maximize their economic performance.

Each of these two parties does not want to share this information with the other, which makes the optimal allocation of resources more difficult. For example, suppliers do not want lay out how many and what kind of machines they have and how they are connected. Even, users don't want to expose their workload details including source codes and sets of their own data to other people, including Suppliers. Therefore, users cannot express their requests for resources so that the allocated resources are optimal, because they do not know exactly how much is available. Similarly, suppliers cannot allocate resources more appropriately to user requests, as there is no information on their workloads.

Cloud computing has made it possible to realize a long-standing dream which was to transform the use computational resources in the form of utility as is the case with electricity. The developers with innovative ideas can launch their businesses without resorting to a large investment capital beforehand. They will not have to worry about over provisioning a service if it is underutilized compared to forecasts

Established beforehand and they will not also have to worry about the opposite scenario. Nevertheless, the confusion remains over exactly what the Cloud is and when it is useful. Indeed, there are many definitions and interpretations of cloud computing which are searchable from different sources. Cloud computing, often referred to simply as "the cloud", is the provisioning in the form on-demand service of IT resources through applications to data centers data via the internet. Payment in the cloud model is pro rata to the use of resources by the consumer, this payment model is called "pay-as-you-go"[32]. In the literature, there are several definitions that are based on different aspects of the cloud, but there has not yet been a consensus.

2. LITERATUR REVIEW
Resource allocation [9], [10] is used to assign available resources in an economic manner. In the field of project management, resource allocation is the planning activities as well as the resources required by these activities while satisfying the constraints associated with it: the limited resource capacity, the project schedule, etc. As part of any

project, resource allocation is the distribution of resources by the system to several components according to their request submits in the plan. In other domain of applications, the allocation of resources is an optimization problem combinatory such as the multi-agent problem, where the resources are the goal of the problem is to find the best allocation of resources to controllers so that all components are controlled as much as possible. All the problems cited above have been addressed centrally, i.e. the computational load required to address the problem was managed by a central entity[33]. So the resolution of the problem of allocation of resources proceeds by a centralized approach that is based on a single central entity. However, in case the computational load of the problem becomes very large, the central entity may not address the problem adequately, because the centralized approach is not a scalable approach. A scalable approach is an approach that can address the resource allocation problem where there are multiple resources are there to allocate and multiple agents are the distributed approach [28]. A distributed approach distributes the computational load of the problem to all agents within the multi-agent system instead of assigning it to a single entity central as is the case for centralized approaches. More specifically, the task of taking decision-making will be distributed to all agents.

The resource allocation problem is a combinatorial optimization problem, where a limited amount of resources should be allocated to a certain number of tasks so that the most efficient allocation of resources is achieved through the optimization of different goals. Resource allocation problem goals are usually to reduce overall cost, processing time, manage conflicts, and maximize profit, effectiveness or compatibility. The resource allocation problem involves usually a huge number of integer variables and multiple objectives in a space of discrete search, which causes their complexity to be NP-complete in the worst case. This nature makes methods exact, like integer programming or linear, inadequate to deal with the problem of resource allocation, and promotes heuristic techniques such as: genetic algorithms (GA) [6], [7], simulated annealing (SA), Differential Evolution (DE) [5], Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO) to obtain approximate solutions.

The simulated annealing (SA) technique was originally proposed to solve the problems related to combinatorial optimization [8]. It is a technique to find a better solution for an optimization problem by trying random variations of the current solution. The main characteristic is that the worst variation can be accepted as a new solution with a probability, this is the major advantage of SA over other optimization techniques. It is often used when the search space is discrete. But in [11], the authors get the initial solution as follows: first they maintain a list of all virtual machines in descending order based on their loads. Then they have map the first virtual machine in the list that has the most workload on a physical machine that has the largest residual capacity, and repeat this process for the next virtual machine until all virtual machines have been allocated to physical servers[34].

Finally, they get an initial solution as input for the algorithm simulated annealing load balancing. Thanks to this initial endowment, they can obtain a pretty good initial possible solution. Genetic algorithms are optimization algorithms based on techniques derived from genetics and natural evolutionary mechanisms: crossover, mutations, selections, etc. The authors in [3] proposed a genetic algorithm (GA) to reach a state of balance between requests and offers to the automatic adjustment of price. GA procedures, GA coding method, GA fitness function and the three operators of GA [14], [15] (Selection operator, Crossing operator, mutation) are described in detail [32]. The Particle Swarm Optimization (PSO) algorithm is a computational method bionic optimization. The algorithm is simple and requires less adjustment of settings. The authors in [39] use the algorithm to adjust the price of different resources respectively. The algorithm has several steps: in the first step the initial price vector for each resource will be defined, and then the PSO algorithm is used to obtain the optimal solution, on the based on the latter the proposed algorithm calculates the total demand of the workloads for each resource. More details in [4], [16].

Cloud computing resources are widely distributed with great diversity. The real-time dynamic changes in user requirements are very difficult to predict accurately. Ant colony algorithm heuristics can be used to solve such problems. The authors in [11] proposed a method to the initialization of the pheromones of each node based on the material resources (power processing, memory capacity, bandwidth, etc. of the virtual machine[35]. The update day of pheromones are carried out at the same time when a new task is assigned to a knot. Based on the formula for calculating the next hop in the classical algorithm of ant colony, the authors proposed a formula to calculate the probability that the ants choosing the next point, among the possible nodes. More details in [11]. Cloud computing systems are characterized by the use of a large pool of computer resources accessible through the Internet on demand [12]. It offers a variety of resources, such as network, storage, and other computing resources to users adopted by IaaS, PaaS, SaaS and other forms of service. These resources are huge, heterogeneous and distributed [13]; Many strategies are used to provide resources to users and they are grouped under the term "strategies for allocating resources". However, the allocation of resources in a cloud environment computing is very complex.

On the other hand, the software agent is a software entity that operates continuously and independent of the given environment, usually associated with other agents for the problem solving. Multi-agent systems have been increasingly attracted by researchers in various fields. We propose in this article to design a multi-agent system for the allocation of resources [20], [21] in cloud computing. Indeed, multi-agent systems adapt well to the design of a resource allocation system where each member must manage and exchange their knowledge and collaborate with others to achieve their goals. In addition, in such open, dynamic and complex environments, there is a need of distributed data, control as well as expertise which makes the use of multi-agent systems beneficial. One of the advantages of Cloud computing lies in the possibility offered to users to carry out parallel calculations. These consist of dividing an application into elementary tasks distributed over several resources that can operate simultaneously[36].

The objective of this distribution is to improve the performance of applications as well as executed compared to sequential execution. Furthermore, the tasks constituting the distributed applications can be linked by constraints such as time and/or data constraints. As a result, the optimal management of available resources and the scheduling of tasks are fundamental aspects in the parallelization of applications. Resource allocation and task scheduling of an application consist in determining the resources to be assigned to each of these tasks and the order in which they should be performed. Resource allocation and task scheduling issues can be classified into two main categories, namely [17][18][19][22][23][24][25]:

Cloud resource modeling defines how to describe infrastructure resources in the cloud. This modeling is essential for all cloud operations, including the management, control and allocation of resources. Algorithms, control and optimization are heavily dependent on resource modeling chosen by the supplier. The services that the cloud will provide to users also depend of this concept. Network and computing resources can be described by several specifications existing ones, such as: Resource Description Framework (RDF) and Network Description Language (NDL). However, in a cloud computing environment, it is very important that the resource modeling takes into account schemas to represent resources virtual machines, virtual networks and virtual applications[37]. Virtual resources must be described in terms of properties and functionalities. The difficulty of matching demand with available resources in the cloud is related to the degree of detail that must be taken into consideration when description of the resources, if the resources are described using a great deal of detail, in particular risk that the resource selection and optimization phase could become difficult and complex to manage. On the other hand, more detail provides more agility and flexibility in resource usage[38]. This concept is called the granularity of the description of the resources.

3. PROPOSED SYSTEM

Resource optimization is a vital task in any cloud-based environment. The figure 2 is the proposed model of cloud-based optimization model using history Kerberos. The proposed system has four major components 1. Pool of cloud request, 2. History Kerberos, 3. Resource Manager, 4. Optimizer and 5. Resource and Process orchestration.

Pool of cloud request: Cloud is mainly meant for distributed computations. The pool of cloud requests contains all the requests generated for the cloud computation.

History Kerberos: The main task of the history Kerberos is a) Tagging b) Appeal resolving c) Logging and d) Notice Providing.

Once the request reached to history Kerberos from request pool, the history Kerberos extracts the unique identifier of the request. The identifier might be a cloud identifier / hardware identifier / user identifier / process identifier etc., the identifier is not the scope of this proposal. Based on the identifier the tag is examined for any past misconduct or any notice has been issued. If any notice has been issued to the request, then the history Kerberos will deny the request and keep pending for some duration of time as a penalty of the notice.

Request ID	Request Message	Cloud Identifier	Notice Tag bits	Appeal bits	Sequence number

Figure 1: Request Message format

The duration of the penalty time depends on the level of notice. The level of notice is identified by its tag bits (2 bits) as given in the figure 1. The tag bits and its penalties are given in following table 1.

Table 1: Tag Penalty chart

S. No.	Notice Tag bits	Description	Penalty Time in ms
1	11	High Risk	1000 second
2	10	Medium Risk	500 ms
3	01	Low Risk	250 ms
4	00	No Risk	No Penalty

If the request does not have any tags or any past issues, then the history Kerberos generates a new tag, and it would be tagged to the request and the request will be forwarded to the resource manager.

The penalty time would be gradually decreased by numbers of attempts. For example, A Request R1 has some past issue, and it has been tagged as "High Risk" (bits 11). The request R1 will be given a penalty of 1 second quarantine. The sequence number would be 000. After the 1 second of quarantine time. The request will be regenerated but the same process. This time the notice bits would still in 11 and the sequence bits is 001. Again, the request R1 would be in the quarantine for 1 second. The process keeps requesting again and again, for every request the sequence bits are incremented until it reached to 111. Then, notice bits are updated to 01 meaning, the request R1 has received the Medium Risk status. The above cycle is repeated until the request R1 gets the status "No Risk". Once the request R1 gets the status "No Risk", it will be forwarded to resource manager for further process. On every time the request gets the status of "No Risk". The appeal bits are updated. More the appeal bits value more the chance of getting denied or the resources attached to the requested process will be preempted. All these activities would be registered in the log files.

Resource Manager: As the name indicates the resource manger, manages all the necessary information regarding the resources it has mainly three compartments or three components namely list of available resources time of allocation and the list of algorithms apart from these three components it has resource gateway and optimizer gateway. The list of available resources contains the unallocated resources which. Time of allocation means it's a log of each resource which have been assigned two which process and the timestamp of the assignment. And the last one is the list of algorithms. The list of algorithms contains the resource management algorithms. the resource manager takes the particular algorithm for the particular strategy to optimize the resources the algorithms it's not the scope of this research.

Optimizer: The optimizer is the main focus of this research article. The resource manager receives the request from the history Kerberos and then forwards to the optimizer. The optimizer extracts the necessary data from the request namely requesting process, unique identifier of the process, tags and ID. As said earlier each request is associated with a tag, ID the optimizer get all the information regarding to their request. Once the optimizer gains the request tag and ID it starts the optimization task based on their level of the notice tag bits. The process may be granted do access the resources or it may be on the hold for some duration until the appeal process completes. The process will gain access of their resources one and only if its notice tag bits come to 00 value. The optimizer employees' resource leveling algorithms, resource smoothing algorithms or reverse resource allocation schemes.

Resource and Process orchestration: Once the optimizer grants the permission to access the resources, then the optimizer links the process to the resource and process orchestration area. Based on the information maintained by the resource manager, the process will be allocated to the requested resources. Otherwise, the process will be held in the resource queue. During the process of utilizing the resources by the process, the resource manager updates the necessary information from the resource and process orchestration area.

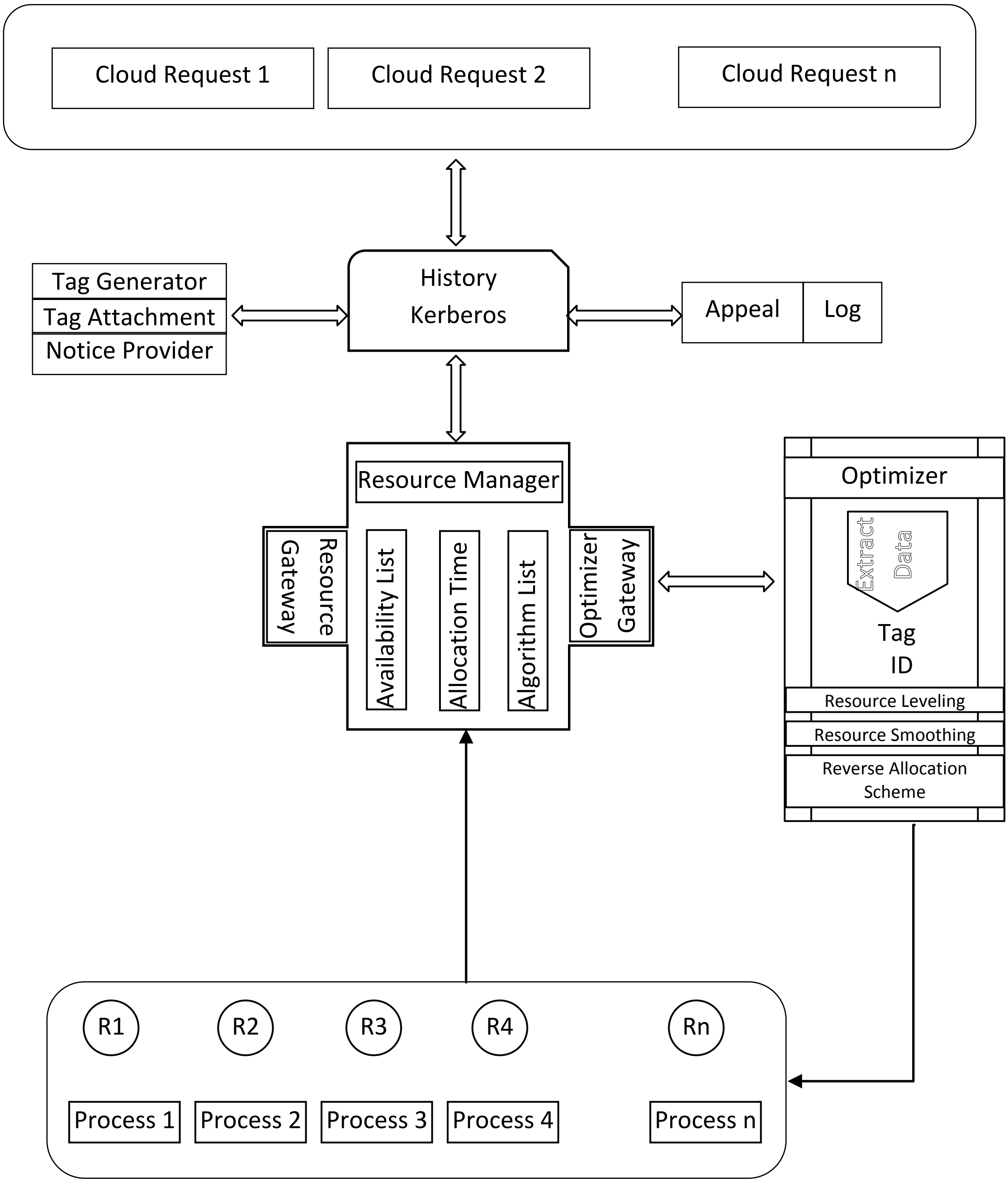

Figure 2: The proposed model of cloud-based optimization model using history Kerberos

4. ANALYSIS

Cloudsim simulator has been used to simulate the proposed algorithm. 20 resources have been taken (R1, R2, …..R20). Each resource is given with a random arrival time, random burst time and Appeal time by the simulator. Below are the four random case scenarios have been shown in Table 2 trough Table 5 with its corresponding figures. For the purpose of explanation, the arrival time, utilization time and response time of R13, R14 and R15 in the Table 1 is same 1.955, 7.454 and 0.665 respectively. But they have different penalties and appeal time, 2, 10 and 6. Hence, R13 got the resources at 8.9 ms, R14 got the resources at 36.66 ms and R15 got the resources at 22.78 ms.

Table 2: Simulator result of random scenario 1

Request / Process	Arrival time	utilization time	response time	waiting time	appeal time	Penalty in ms	Resource allocation time
R1	0.557	1.635	0.243	2.609	10	0.765	26.64
R10	0.604	1.776	0.264	2.833	8	0.831	23.27
R6	0.756	2.219	0.331	3.542	4	0.039	14.92
R8	0.756	2.219	0.331	3.542	4	0.039	14.92
R9	0.756	2.219	0.331	3.542	9	0.039	32.63
R11	0.782	2.982	0.266	1.388	4	0.978	6.33
R20	0.833	3.175	0.283	1.478	5	0.041	8.22
R2	0.835	2.452	0.365	3.913	4	0.148	16.49
R7	1.007	2.959	0.441	4.722	4	0.385	19.90
R16	1.041	3.968	0.354	1.848	9	0.301	17.67
R18	1.041	3.968	0.354	1.848	10	0.301	19.52
R19	1.041	3.968	0.354	1.848	9	0.301	17.67
R12	1.173	4.472	0.399	2.082	6	0.466	13.67
R17	1.388	5.291	0.472	2.463	3	0.735	8.78
R3	1.391	4.087	0.609	6.522	10	0.913	66.61
R4	1.391	4.087	0.609	6.522	7	0.913	47.04
R5	1.391	4.087	0.609	6.522	6	0.913	40.52
R13	1.955	7.454	0.665	3.470	2	0.856	8.90
R14	1.955	7.454	0.665	3.470	10	0.965	36.66
R15	1.955	7.454	0.665	3.470	6	0.444	22.78
						Aveg.	23.16

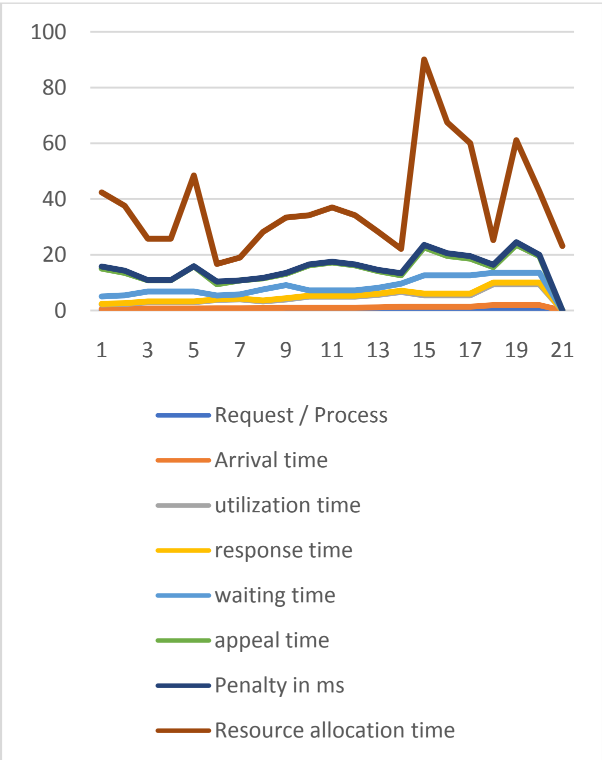

Figure 3: Simulator result of random scenario 1

Table 3: Simulator result of random scenario 2

Request / Process	Arrival time	utilization time	response time	waiting time	appeal time	Penalty in ms	Resource allocation time
R1	0.557	1.635	0.243	2.609	2	0.765	5.77
R10	0.604	1.776	0.264	2.833	2	0.831	6.27
R6	0.756	2.219	0.331	3.542	3	0.039	11.38
R8	0.756	2.219	0.331	3.542	5	0.039	18.46
R9	0.756	2.219	0.331	3.542	9	0.039	32.63
R11	0.782	2.982	0.266	1.388	5	0.978	7.72
R20	0.833	3.175	0.283	1.478	6	0.041	9.70
R2	0.835	2.452	0.365	3.913	9	0.148	36.05
R7	1.007	2.959	0.441	4.722	9	0.385	43.51
R16	1.041	3.968	0.354	1.848	1	0.301	2.89
R18	1.041	3.968	0.354	1.848	3	0.301	6.58
R19	1.041	3.968	0.354	1.848	1	0.301	2.89
R12	1.173	4.472	0.399	2.082	3	0.466	7.42
R17	1.388	5.291	0.472	2.463	5	0.735	13.71
R3	1.391	4.087	0.609	6.522	9	0.913	60.09
R4	1.391	4.087	0.609	6.522	2	0.913	14.43
R5	1.391	4.087	0.609	6.522	9	0.913	60.09
R13	1.955	7.454	0.665	3.470	10	0.856	36.66
R14	1.955	7.454	0.665	3.470	3	0.965	12.37
R15	1.955	7.454	0.665	3.470	10	0.444	36.66
						Aveg.	21.26

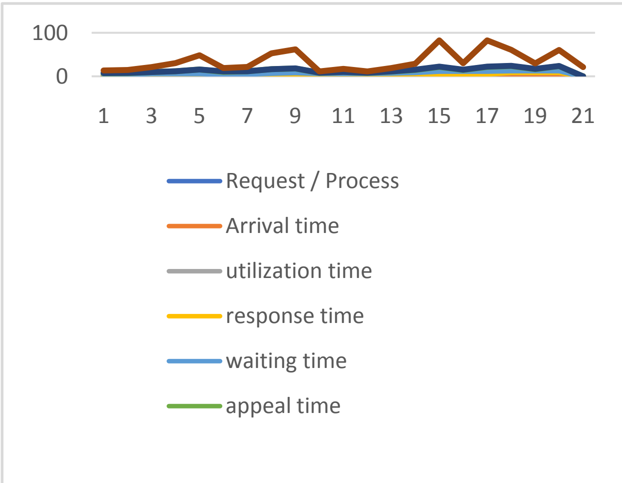

Figure 4: Simulator result of random scenario 2

Table 4: Simulator result of random scenario 3

Request / Process	Arrival time	utilization time	response time	waiting time	appeal time	Penalty in ms	Resource allocation time
R1	0.557	1.635	0.243	2.609	8	0.765	21.43
R10	0.604	1.776	0.264	2.833	4	0.831	11.94
R6	0.756	2.219	0.331	3.542	8	0.039	29.09
R8	0.756	2.219	0.331	3.542	10	0.039	36.17
R9	0.756	2.219	0.331	3.542	1	0.039	4.30
R11	0.782	2.982	0.266	1.388	2	0.978	3.56
R20	0.833	3.175	0.283	1.478	8	0.041	12.66
R2	0.835	2.452	0.365	3.913	3	0.148	12.57
R7	1.007	2.959	0.441	4.722	9	0.385	43.51
R16	1.041	3.968	0.354	1.848	5	0.301	10.28
R18	1.041	3.968	0.354	1.848	1	0.301	2.89
R19	1.041	3.968	0.354	1.848	10	0.301	19.52
R12	1.173	4.472	0.399	2.082	6	0.466	13.67
R17	1.388	5.291	0.472	2.463	5	0.735	13.71
R3	1.391	4.087	0.609	6.522	1	0.913	7.91

R4	1.391	4.087	0.609	6.522	2	0.913	14.43
R5	1.391	4.087	0.609	6.522	10	0.913	66.61
R13	1.955	7.454	0.665	3.470	7	0.856	26.25
R14	1.955	7.454	0.665	3.470	10	0.965	36.66
R15	1.955	7.454	0.665	3.470	3	0.444	12.37
						Aveg.	19.97

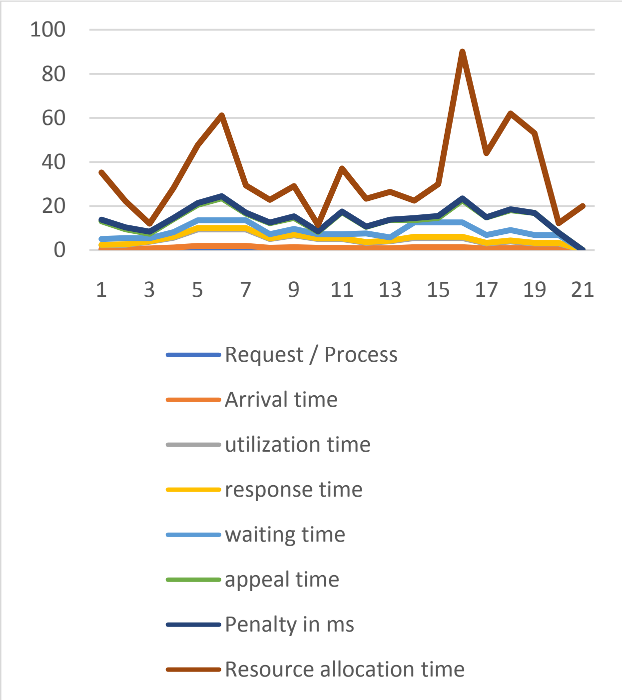

Figure 5: Simulator result of random scenario 3

Table 5: Simulator result of random scenario 4

Request / Process	Arrival time	utilization time	response time	waiting time	appeal time	Penalty in ms.	Resource allocation time
R1	0.557	1.635	0.243	2.609	10	0.765	26.64
R10	0.604	1.776	0.264	2.833	4	0.831	11.94
R6	0.756	2.219	0.331	3.542	7	0.039	25.55
R8	0.756	2.219	0.331	3.542	6	0.039	22.00
R9	0.756	2.219	0.331	3.542	8	0.039	29.09
R11	0.782	2.982	0.266	1.388	5	0.978	7.72
R20	0.833	3.175	0.283	1.478	2	0.041	3.79
R2	0.835	2.452	0.365	3.913	4	0.148	16.49
R7	1.007	2.959	0.441	4.722	8	0.385	38.78
R16	1.041	3.968	0.354	1.848	4	0.301	8.43
R18	1.041	3.968	0.354	1.848	5	0.301	10.28
R19	1.041	3.968	0.354	1.848	8	0.301	15.82
R12	1.173	4.472	0.399	2.082	9	0.466	19.91
R17	1.388	5.291	0.472	2.463	1	0.735	3.85
R3	1.391	4.087	0.609	6.522	2	0.913	14.43
R4	1.391	4.087	0.609	6.522	4	0.913	27.48
R5	1.391	4.087	0.609	6.522	4	0.913	27.48
R13	1.955	7.454	0.665	3.470	8	0.856	29.72
R14	1.955	7.454	0.665	3.470	1	0.965	5.43
R15	1.955	7.454	0.665	3.470	6	0.444	22.78
						Avg.	18.38

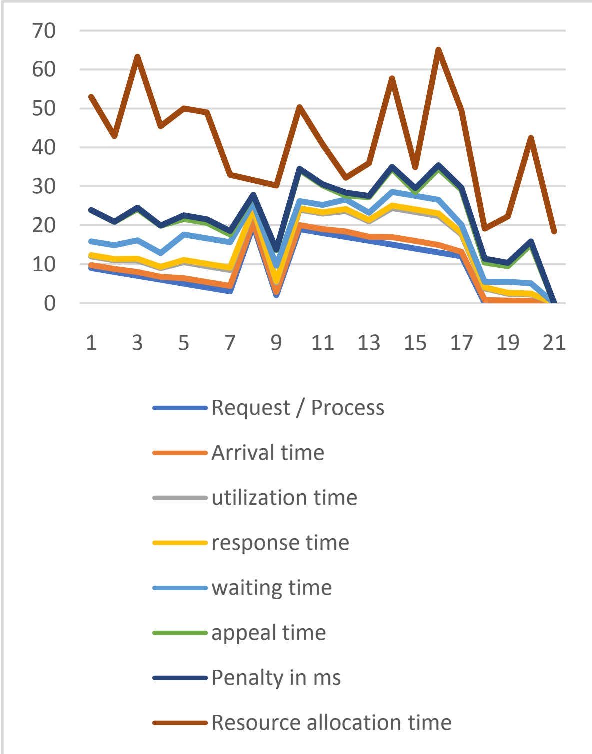

Figure 6: Simulator result of random scenario 4

5. CONCLUSION

The proposed cloud-based resource allocation yields the significant result with the help of history Kerberos. The definition of new request format that is shown in figure 1, has played a vital role in identifying the nature of the request and to identify the penalties associated with the requested process. The penalty time and the decrement of the appeal cycle may be varied and must follow the policy of resource management. Unauthorized request can be dealt with any security algorithms to make the system rigid and safe. The issues and misconduct of the requisition system or process also defined in the resource allocation policy.

6. REFERENCES

[1] Bowman, M., Debray, S. K., and Peterson, L. L. 1993. Reasoning about naming systems.

[2] G. Lee. (2012). “Resource Allocation and Scheduling in Heterogeneous Cloud Environments”, Electrical Engineering and Computer Sciences University of California at Berkeley, Technical Report No. UCB/EECS-2012-78.

[3] Nasar, M., Johri, P., & Chanda, U. (2014). Resource Allocation Policies for Fault Detection and Removal Process. International Journal of Modern Education and Computer Science, 6(11), 52.

[4] Y. Xindong ET al. (2011). “ARAS-M: Automatic Resource Allocation Strategy based on Market Mechanism in Cloud Computing”, Journal of computers.

[5] F. Xie ET al. (2013). “A Resource Allocation Strategy Based on Particle Swarm Algorithm in Cloud Computing Environment”, Fourth International Conference on Digital Manufacturing and Automation (ICDMA).

[6] Md. Nasar, Prashant Johri and Udayan Chanda. (2013). "A Differential Evolution Approach for Software Testing

Effort Allocation," Journal of Industrial and Intelligent Information, Vol. 1, No. 2, pp. 111-115, doi: 10.12720/jiii.1.2.111-115.

[7] Johri, P., Nasar, M., Chanda, U. (2013). "A genetic algorithm approach for optimal allocation of software testing effort". International Journal of Computer Applications. 68, 21–25.

[8] Md. Abu Kausar, Md. Nasar, and Sanjeev Kumar Singh. (2013). "A Detailed Study on Information Retrieval using Genetic Algorithm," Journal of Industrial and Intelligent Information, Vol.1, No.3, pp. 122-127, doi: 10.12720/jiii.1.3.122-127.

[9] F. Zongqin ET al. (2013). "Simulated-Annealing Load Balancing for Resource Allocation in Cloud Environments", International Conference on Parallel and Distributed Computing, Applications and Technologies.

[10] Nasar, M., Johri, P. (2014). "Testing and Debugging Resource Allocation for Fault Detection and Removal Process". International Journal of New Computer Architectures and their Applications, no. 4, pp. 193—200.

[11] Nasar, M., & Johri, P. (2015). Testing Resource Allocation for Modular Software using Genetic Algorithm. IJNCAA, Vol. 5, No. 1, pp. 29-38.

[12] Y. Zhengqiu ET al. (2012). "Study on cloud resource allocation strategy based on particle swarm ant colony optimization algorithm", proceedings of IEEE (CCIS).

[13] R. Buyya, R. Ranjan and R. N. Calheiros. (2010). "InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services", Algorithms and Architectures for Parallel Processing -10th International Conference-(ICA3PP).

[14] X. Wang, X. Liu et al. (2013). "A Decentralized Virtual Machine Migration Approach of Data Centers for Cloud Computing", Hindawi Publishing Corporation Mathematical Problems in Engineering Volume.

[15] Nasar M, Johri P, Chanda U. (2014). "Software testing resource allocation and release time problem: a review". International Journal of Modern Education and Computer Science 6(2):48-55

[16] P. Johri, M. Nasar, S. Das, and M. Kumar. (2016). "Open source software reliability growth models for distributed environment based on component-specific testing-efforts". In Proceedings of the 2nd International Conference on Information and Communication Technology for Competitive Strategies. 75. DOI:10.1145/2905055.2905283

[17] Nasar, Md., Johri, P., Chanda, U. (2014). "Dynamic effort allocation problem using genetic algorithm approach". International Journal of Modern Education and Computer Science 6(6), 46–52

[18] S. G. Kumar, C. Shin Yeo, A. Anandasivam and R. Buyya, (2011). Environment conscious scheduling of HPC applications on distributed Cloud oriented data center, Journal of Parallel and Distributed Computing (71 :6), pp. 732 -749.

[19] E. Juhnke, T. Dornemann, D. Bock, and B. Freisleben,, (2011). "Multi-objective Scheduling of BPEL Workflows in Geographically Distributed 186 Clouds", IEEE International Conference on Cloud Computing (CLOUD), pages 412-419.

[20] K. Boloor, R. Chirkova, Y. Viniotis, and T. Salo, (2010). "Dynamic Request Allocation and Scheduling for Context Aware Applications Subject to a Percentile Response Time SLA, in a Distributed Cloud", IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), pages 464-472.

[21] M. Nasar and P. Johri. (2016). "Testing resource allocation for fault detection process". In Smart Trends in Information Technology and Computer Communications. A. Unal et al. (Eds.). 683--690. DOI:10.1007/978-981-10-3433-6_82.

[22] Md. Abu Kausar, Md. Nasar & Sanjeev Kumar Singh. (2013). "Information Retrieval using Soft Computing: An Overview", International Journal of Scientific & Engineering Research, Vol. 4, Issue. 4.

[23] D. Niyato, A.V. Vasilakos, and K. Zhu. (2011). Resource and Revenue Sharing with Coalition Formation of Cloud Providers: Game Theoretic Approach, 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pages 215-224.

A. Oprescu, T. Kielmann, (2010). "Bag-of-Tasks Scheduling under Budget Constraints", IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), pages 351-359.

[24] F. Zhang, J. Cao, K. Hwang, and C. Wu. (2011). "Ordinal Optimized Scheduling of Scientific Workflows in Elastic Compute Clouds", In Proceedings of the 2011 IEEE Third International Conference on Cloud Computing Technology and Science.

[25] R. Van Bossche, K. Vanmechelen, and J. Broeckhove. (2011). Cost-Efficient Scheduling Heuristics for Deadline Constrained Workloads on Hybrid Clouds, IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), pages 320-327.

[26] Ejarque J. (2010). "A Multi-agent Approach for Semantic Resource Allocation". 2010 IEEE Second International Conference on Cloud Computing Technology and Science, pp. 335- 342.

[27] Endriss U., Maudet N., Sadri F., and Toni F. (2003). "On optimal outcomes of negotiations over resources". In Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2003). ACM Press.

[28] Endriss U., Maudet N., Sadri F., and Toni F., (2006). Negotiating socially optimal allocations of resources. Journal of Artificial Intelligence Research, 25:315–348.

[29] Endriss U. and Maudet N. (2005). "On the communication complexity of multilateral trading: extended report". Journal of Autonomous Agents and Multi-agent Systems 11, 1, 91– 107.

[30] Dr. Saleem Basha, Mohamed Yasin Noor Mohamed and P. Sujatha, "System Design of MEB in M-IWD Model with Heuristic Function on WSN", Recent Patents on Engineering, Bentham Science Publisher, Vol 14(4), 2020.

[31] Amin Mohammed, Saravana Balaji, Dr. M.S. Saleem Basha and Asha, "FCO - Fuzzy constraints applied Cluster Optimization technique for Wireless AdHoc Networks",

International Journal of Computer Communications, Vol 154, pp 501-508, 2020, ISSN: 01403664, Elsevier.

[32] Dr. M.S. Saleem Basha, N. Moganarangan, "A novel algorithm for reducing energy-consumption in cloud computing environment: Web service computing approach", Journal of King Saud University - Computer and Information Sciences, Vol 28(1), pp 55-67, 2016, ISSN: 13191578.

[33] Dr. M.S. Saleem Basha, J. Amudhavel, "A Comprehensive Analysis on Multi Agent Decision Making Systems", Indian journal of Science and technology, Vol 9(11), pp 1-2, 2016, ISSN: 0974-5645.

[34] C.S. Cilpa, Dr. M.S. Saleem Basha, "A Comparative Analysis of Scheduling Policies in Cloud Computing Environment", International Journal of Computer Applications, Vol. 67(20), pp 16-24, 2013, ISSN: 0975-8887, NASA ADS.

[35] M.Shanmugam, M.S. Saleem Basha, P.Dhavachelvan, "A Study on Communication Protocols and Applications in VANET", International Journal of Applied Engineering Research, Vol. 8(10), pp 1185 - 1204, 2013, ISSN: 0973-4562.

[36] S. Venkatesan, M.S. Saleem Basha, C. Chellappan, Anurika Vaish, P. Dhavachelvan, "Analysis of accounting models to detect duplicate requests in web service", International Journal of King Saud University – Computer and Information Sciences, Vol. 25(1), pp 7-24, 2013, ISSN: 1319-1578.

[37] M.Shanmugam, M.S. Saleem Basha, "DDoS Attack Traceback and Chaos in a Distributed Network a Survey", International Journal of Applied Engineering Research, Vol 8(10), pp 1159 - 1169, 2013, ISSN: 0973-4562.

[38] J.Amuthavel, M.S. Saleem Basha, P.Dhavachelvan, R. Baskaran, "Efficient optimal Packet Management in Distributed Wireless Ad-Hoc Environments Using DST", International Journal of Engineering Science and Technology, vol. 2(6), pp 2278-2287, 2010, ISSN 0975-5462.

[39] M. Sathya, P.Venil, M.S. Saleem Basha, "A Novel Approach to Concept Extraction Using Naive Bayesian Classification Technique" International Journal of Soft Computing, vol. 2 (4), pp 488-493, 2007, ISSN: 1816-9503