Effective Resource Management in Clouds Using Advance Reservation

Gayatri S, Madhuri Bhavsar
M.Tech Computer Engineering Department
Nirma Institute of Technology (NIT),
Ahmedabad, India
gayatri_jain@rediffmail.com

Abstract—**Advance Reservation (AR) for Clouds is now a significant research focus as it allows consumers to gain synchronized access for their applications to be executed in parallel, and assures the availability of resources at specified future times. Clouds could be leased by the faculties of the college, who want to demonstrate the execution of the parallel program on a distributed system during the lab hours or if a firm wants to lease the web server for months or years, a dedicated server in a data center may be from Amazon EC2 [17]. The Resource provisioning model proposed in this paper provides a support for immediate, best-effort provisioning and AR, including preempting Virtual Machines (VM) in the favor of an AR, suitable for the above specified scenarios. AR of Resources offers a better Quality of Service (QoS) for time critical applications. Here, an effort is also made to discuss the Parameters of the Reservation and the possible states the Reservation made by a consumer may be in. We also show how the ARM (Advance Resource Management) module can be embedded in the general architecture of the Cloud.**

Keywords: Cloud, Advance Reservation, Synchronized Access, Active, Commit.

I. INTRODUCTION

"A Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers."[1]. In most of the Clouds (like Eucalyptus [16], OpenNebula [8] and Amazon EC2 [17]), the request placed by the consumer is initially placed in a Queue Data Structure if there are no required resources available or scheduled immediately if available. Thus, there would be no assurance as to when the request would be executed.

This could be a major issue in parallel and distributed applications where most of the processes are dependent on each other. Such processes could be handled effectively by AR. AR in a Cloud allows consumers to achieve synchronized access for their applications to be executed in parallel, and assures the availability of requested resources to the consumer at specified future times. Clouds could be leased by the faculties of the college, who want to demonstrate the execution of the parallel program on a distributed system during the lab hours or if a firm wants to lease the web server for months or years, a dedicated server in a data center may be from Amazon EC2 [17]. The Resource provisioning model proposed in this paper provides a support for immediate, best-effort provisioning and AR, including preempting Virtual Machines (VM) in the favor of an AR if required and is suitable for the above-specified scenarios. Common resources which could be requested or leased are Nodes (VM), network bandwidth, memory and disk space.

The later part of this paper is structured as follows: Section 2 discusses Related Work, Section 3 discusses the Resource provisioning model, Section 4 discusses the Reservation Request Model, Section 5 discusses the States of Advance Reservation and Section 6 Concludes the paper.

II. RELATED WORK

Quite a few Research Clusters have worked in the related areas like Cluster, Grid and Cloud, addressing varied topics like Job Scheduling on virtual clusters [3, 4, 5, 6], but all of these focus on meeting the requirements of a single provisioning situation (either best effort or immediate but no ARs) except for Walters et al [6] and Borja Sotomayor et al [2, 8].

Resource management in Clouds is at a finer granularity and has many related levels. Managing resources dynamically and agilely in terms of the varied requirements of consumers is a challenge in Cloud Computing environments, and a Resource Allocation Strategy based on Market Mechanism (RAS-M) is proposed to settle this problem. RAS-M tries its best to achieve the equilibrium state through employing the present GA-based price adjusted algorithm [10].

OpenNebula is an open source virtual infrastructure manager that can be used to deploy virtualized services on both, a local pool of resources and external IaaS (Infrastructure as a Service) clouds. Haizea, a resource lease manager, can act as a Scheduling backend for OpenNebula providing features not found in other cloud software or virtualization-based datacenter management software, such as advance reservations and resource preemption [8].

The model proposed in this paper provides a support for immediate, best-effort provisioning and AR, including preempting Virtual Machines in the favor of an AR if required. AR guarantees the availability of resources to the consumers and applications at the requested times and offers better QoS. It also specifies the parameters of the Reservation and the possible states in which the Reservation made by a consumer can be in.

III. RESOURCE PROVISIONING MODEL

The model depicted in Fig 1 [18] shows how ARM is embedded in the general architecture of the cloud and how it interacts with the other components. The general architecture consists of the following components:

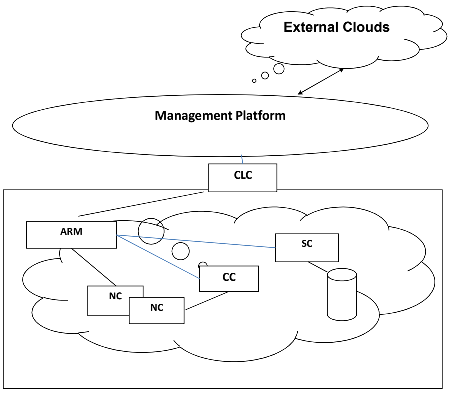

Figure 1. ARM embedded in the general Architecture of the Cloud.

Cloud Controller (CLC) is the access-point into the cloud for administrators, project managers, developers and end-users. It is responsible for querying the node managers for information about resources and making high level decisions for scheduling and implementing the above decisions by making requests to cluster controllers.

Node Controller (NC) is executed on every node that is designated for hosting VM instances. NCs control the execution, inspection, and termination of VM instances on the host where it fetches, runs and cleans up local copies of instance images (the kernel, the root file system and the ram disk image). It also queries and controls the system software on its node (host OS and the hypervisor) in response to queries and controls requests from the cluster controller. The Node controller is also responsible for the management of the virtual network endpoint.

Cluster Controller (CC) generally executes on a cluster front-end machine or any machine that has network connectivity to both the nodes running NCs and to the machine running the CLC. It collects information about a set of VMs and schedules them for execution on specific NCs. It also manages the virtual instance network and participates in the enforcement of SLAs (Service Level Agreements) as directed by the CLC. All nodes served by a single CC must be in the same broadcast domain (Ethernet).

Storage Controller (SC) implements block-accessed network storage. An Elastic Block Store (EBS) is a Linux block device that can be attached to a virtual machine but sends disk traffic across the locally attached network to a remote storage location. An EBS volume cannot be shared across instances but does allow a snap-shot to be created and stored in a central storage system.

Management Platform provides an interface to various cloud services and modules. These features can include VM management, user/group management, accounting, monitoring, storage management, cloud-bursting, SLA definition, and enforcement provisioning.

ARM, the Resource provisioning model, is the model that is responsible for Advance Reservation. It interacts with NC and CLC before confirming the reservation to the consumer. It also provides support for immediate and best-effort provisioning.

IV. RESERVATION REQUEST PARAMETERS

Initially, a reservation request consists of the <N_q, D_m, T_q, R_t, RT, RSP, A> where N_q is the Number of nodes desired, D_m is the Duration, the amount of time the Reservation will last, T_q is the desired Start time, R_t any other Resource Types, RT is a particular type of Reservation, RSP is the Resource Specific Parameters (parameters that are unique to each type of resource, such as bandwidth for a network reservation and number of nodes for a computational reservation) and A is the Application to run on those resources. Further, an assumption is made that the following procedure occurs when a consumer wishes to submit a reservation request:

1) The consumer queries if he can run an application at time T_q (the start time) on N_q the Number nodes for at most T_m amount of time.

2) The scheduler makes the reservation for the consumer at time T_q if it can. In this case, the reservation time, T is equal to the requested reservation time, T_q.

3) If the scheduler cannot make the reservation at time T_q, it replies with a list of possible times it could make the reservation, and the consumer picks the available time T which is flanking in time to T_q.

The final part of the model is what occurs when an application is terminated. Primarily, only applications that come from a queue can be terminated. Subsequently, when an application is terminated, it is placed back in the queue from which it comes in its proper position.

V. STATES OF ADVANCE RESERVATION

An Advance Reservation request can be in one of several states during its lifetime as shown in Fig 2[7]. Transitions between the states are defined by the operations that a consumer performs on the reservation. These states are defined as follows:

Request: When a request for a reservation of Resources is first made, it is in the Initial state.

Reject: The reservation is not fruitfully allocated; cause may be the existing reservation has expired or the slots may be full.

Accept: A request for a new reservation has been agreed upon.

Commit: A reservation has been confirmed by a consumer before the expiry time and will be privileged by the requested Resource.

Change Request: A consumer is trying to alter the requirements for the reservation prior to its starting. If it is successful, the reservation is committed with the new requirements; otherwise, the parameters remain the same. For instance, one can increase the bandwidth that has already been requested. A modification that reduces its requirements normally succeeds, although certain factors may cause reduction modifications to fail, such as local policy that does not allow small reservations on some resources. In no case should the underlying system implement a modification such that if the modification fails, the original reservation is lost. For example, a simple implementation may implement modification by canceling a reservation and making a new reservation, but if the new reservation fails and it cannot be rolled back to the original reservation, this would be undesirable.

Active: The system executes the reservation as the start time has been reached.

Cancel: A consumer may cancel the reservation as it may no longer be required.

Complete: The reservation's end time has been reached.

Terminate: A consumer terminates an active reservation before the end time is reached.

Bind Reservation*:* When the application is ready to use a reservation, it may need to provide run-time information that was not available at the time the reservation was made as shown in Fig. 3. This is known as *binding* a reservation [9]. For example, network reservations require port numbers to be specified, but they are not usually known at reservation time. Not all reservations require such run-time parameters.

Unbind Reservation: A reservation can be unbound. It then will no longer be usable by the person using the reservation as shown in Fig. 3. It can be rebound, however with new run-time parameters as shown in Fig. 4.

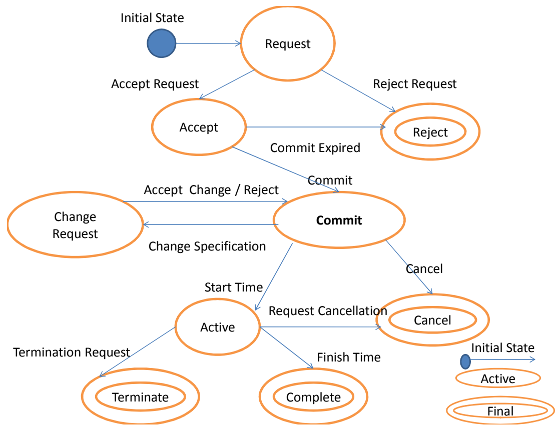

Figure 2. A State Transition diagram depicting the states for Advance Reservation

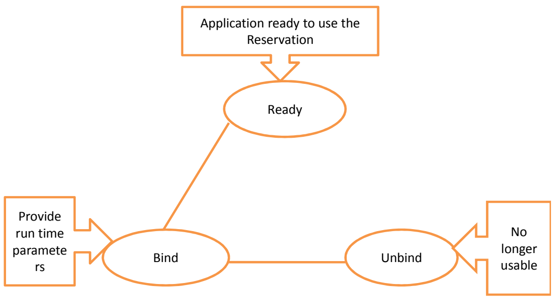

Figure 3. A State Transition diagram depicting Bind and Unbind

Figure 4. A State Transition diagram depicting Bound and Rebound

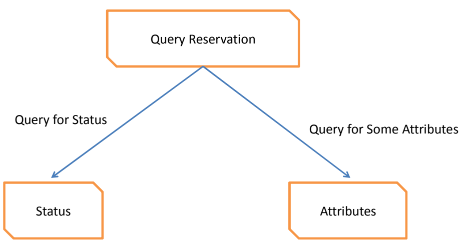

Figure 5. Diagram depicting Query Operation

Query Reservation Status: Consumer can query for the status of a reservation by polling it. The status includes whether the start of the reservation has begun and whether the reservation has been committed as shown in Fig. 5.

Query Reservation Attributes: Consumer can query for attributes associated with an existing reservation. These include begin and end time of the given reservation and whether it is a two-phase commit reservation. The attributes also include specific information required to actually use a reservation. For example, attributes are a Folder name where data was staged on or a Queue name which has to be used for submitting a job as shown in Fig. 5.

Register Callback: One can provide a function that will be called when the status of a reservation changes or when the reservation manager wishes to provide extra information to the application. This information may include notification that the related reservation appears to be too small.

VI. Conclusion

This paper proposes a Resource provisioning model for cloud that provides a support for immediate, best-effort provisioning and ARs, including preempting VM in the favor of an AR if required and is suitable for the specified scenarios where lease would turn cheaper than installation of the resources. Common resources which could be requested or leased are Nodes (VM), network bandwidth, memory and disk space. AR in a Cloud allows consumers to achieve synchronized access for their applications to be executed in parallel and assures the availability of requested resources to the consumer at specified future times. With AR thus resources could be managed effectively and would offer a better QoS.

References

[1] Rajkumar Buyya1,2, Chee Shin Yeo1, and Srikumar Venugopal and Grid Computing and Distributed Systems (GRIDS) Laboratory Department of Computer Science and Software Engineering. Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities. In the Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications: page 5-13

[2] Borja Sotomayor ,Rub´en Santiago Montero Ignacio Mart´ın Llorente, Ian Foster. Resource Leasing and the Art of Suspending Virtual Machines, In the Proceedings of 2009 11th IEEE International Conference on High Performance Computing and Communications: page 59 – 68.

[3] N. Fallenbeck, H.-J. Picht, M. Smith, and B. Freisleben. Xen and the art of cluster scheduling. In *VTDC '06: Proceedings of the 1st International Workshop on Virtualization Technology in Distributed Computing*. IEEE Computer Society, 2006.

[4] W. Emeneker and D. Stanzione. Efficient Virtual Machine Caching in Dynamic Virtual Clusters. In *SRMPDS Workshop, ICAPDS 2007 Conference*, December 2007.

[5] N. Kiyanclar, G. A. Koenig, and W. Yurcik. Maestro VC: A paravirtualized execution environment for secure on-demand cluster computing. In *CCGRID '06: Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID' 06)*, page 28. IEEE Computer Society, 2006.

[6] J. P. Walters, B. Bantwal, and V. Chaudhary. Enabling interactive jobs in virtualized data centers. In *Cloud Computing and Applications 2008 (CCA08)*, 2008.

[7] Anthony Sulistio and Rajkumar Buyya : A GRID SIMULATION INFRASTRUCTURE SUPPORTING ADVANCE RESERVATION

[8] Borja Sotomayor, Rub´en S. Montero, Ignacio M. Llorente, and Ian Foster: An Open Source Solution for Virtual Infrastructure Management in Private and Hybrid Clouds, IEEE INTERNET COMPUTING, SPECIAL ISSUE ON CLOUD COMPUTING July 2009

[9] Alain Roy Scheduling Working Group University of Wisconsin-Madison,Forschungszentrum Jülich GmbH May 2002

[10] Xindong YOU, Xianghua XU, Jian Wan, Dongjin YU School of Computer Science and Technology Hangzhou Dianzi University Hangzhou, China,youxindong@hdu.edu.cn, wanjian@hdu.edu.cn: ***RAS-M:Resource*** *Allocation Strategy based on Market Mechanism in CloudComputing* 2009 IEEE page 256 – 253

[11] Scheduling with Advanced Reservations Warren Smith_y Ian Foster_ Valerie Taylory _Mathematics and Computer Science Division Argonne National Laboratory, Argonne, IL 60439 fwsmith,fosterg@mcs.anl.gov

[12] Lizhe Wang, Jie Tao, Marcel Kunze Institute for Scientific Computing, Research Center Karlsruhe Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany Alvaro Canales Castellanos, David Kramer, Wolfgang Karl Department of Computer Science, University Karlsruhe (TH) 76128 Karlsruhe, Germany : Scientific Cloud Computing: Early Definition and Experience ,In the proceedings of the 10th IEEE International Conference on High Performance Computing and Communications, page 825 – 830

[13] Borja Sotomayor, Kate Keahey, Ian Foster : Combining Batch Execution and Leasing Using Virtual Machines, HPDC'08, June 23–27, 2008, Boston, Massachusetts, USA, ACM.

[14] Luqun Li An Optimistic Differentiated Service Job Scheduling System for Cloud Computing Service Users and Providers: 2009 Third International Conference on Multimedia and Ubiquitous Engineering. Page 295 - 299

[15] Massimiliano Rak , Emilio P. Mancini,UmbertoVillano PerfCloud: GRID Services for Performance-oriented Development of Cloud Computing Applications, In 2009 18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises ,page 201 – 206

[16] Eddy Caron, Frederic Desprez, David Loureiro , Adrian Muresan Cloud Computing Resource Management through a Grid Middleware: A Case Study with DIET and Eucalyptus ,in the proceedings of 2009 IEEE International Conference on Cloud Computing, page 151 – 154

[17] Amazon Elastic Compute Cloud. http://aws.amazon.com/ec2/.

[18] http://www.eucalyptus.com/