Cost-Effective Computing in the Cloud Infrastructure

Mohammad Razavi and Kamran Zamanifar

***Abstract*—Virtualization can provide substantial benefits to cloud infrastructure services, also known as Infrastructure as a Service (IaaS), by facilitating dynamic resource management. We proposed an integrated architecture to extend the cloud infrastructure services with live resource scaling and migration of virtual machines to achieve cost-effective computing in IaaS. A prototype of the proposed architecture for memory scaling of the KVM virtual machines with the promising results is also implemented in Eucalyptus open-source private cloud platform.**

***Index Terms*—Dynamic resource management, virtual machine scaling, migration, IaaS.**

I. Introduction

Today, millions of web sites and services are hosted on virtual machines which are usually referred to as virtual private servers (VPS) by internet hosting service providers. A budget 384 MB VPS could handle 1000's of concurrent users [4] by means of operating system-level virtualization i.e. a virtualization technique which virtualizes servers by sharing resources of a physical server between several operating systems.

Cloud infrastructure services, also known as infrastructure as a service (IaaS), deliver hardware infrastructure—typically a virtual server—as a service and make employing servers more convenient and cost-effective as the definition of cloud computing by NIST is "a pay-per-use model for enabling available, convenient and on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."

A public cloud e.g. Amazon Elastic Compute Cloud (EC2) provides IaaS available to the general public over internet, while a private cloud provides such services for a single organization and usually is managed internally. Hence, the cloud infrastructure must not be very different. The schematic of such cloud with infrastructure services is depicted in Fig. 1. Most of today's clouds with IaaS have almost the same organization [5] and they solely feature creating and removing instances of virtual machines to handle users' on-demand computing resources. Although most current hypervisors have online resource scaling capabilities, users can only determine resources bound to each virtual machine on instance creation time usually with the exception of disk and I/O resources. Hence, in the following, the term "resource" refers to all computing resources (e.g. memory, CPU and network bandwidth) with the exception of I/O resources.

In the scheme illustrated in Fig. 1, each user has to determine his maximum resource requirements in a certain period of time and calls the proper web service of the cloud to create a new virtual machine to satisfy his requirements. Hence, if a computing task like most web services requires availability for a long period of time, although they might use the maximum resource capacity for a short period, the maximum required resources has to be requested for the entire time. Therefore, the challenge is providing better resource scaling features for the cloud infrastructure with an effective resource management mechanism.

The rest of this paper is organized as follows. After introducing related works in section II the proposed architecture to extend IaaS resource scaling capabilities is described in section III while section IV illustrates our experiments with Eucalyptus private cloud. Finally, in section V our conclusions are presented.

II. Related Works

SnowFlock [1] address infrastructure resource management problems by providing VM fork. As Lagar-Cavilla et al. describe, lack of a general resource management mechanism forces users of cloud computing into ad hoc practices to manage application states for their resource scaling requirements. VM fork is the cloning of a virtual machine into multiple replicas running on different hosts. Hence users, instead of creating a new instance of the virtual machine and restoring it to the proper state, could rapidly clone an already existing VM.

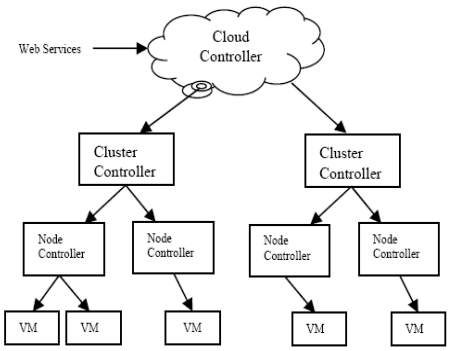

Fig. 1. Cloud IaaS schematic

Jong-Guen Park et al. [2] describe challenges that come with server virtualization as increasing resource utilization while satisfying the service level objectives of software services running on them. To achieve an effective resource management of virtual servers they suggest either online resource scaling or migrating virtual machines from a host

Manuscript received August 20, 2012; revised November 18, 2012.

The authors are with Computer Engineering Department, University of Isfahan (e-mail: razavi@eng.ui.ac.ir, zamanifar@eng.ui.ac.ir).

physical machine to another. Finally, they propose an optimization model based on linear programming for virtual machine migration in a self-managing virtualized server environment.

Merwe et al. [3] consider cloudbursting and follow-the-sun use cases in cloud environment and they conclude that the real challenge in cloud infrastructure is finding a service abstraction with a balance between user complexity and cloud provider complexity when managing resources with a holistic architecture.

III. The Proposed Architecture

Our proposed architecture extends cloud platform capabilities by adding a dynamic resource scaling module to its cluster controller. Each physical machine in the cluster has limited resources, thus to be able to scale its VMs resources it has to leave some of its resources unutilized. Another problem is that there is no way to guarantee a fixed amount of resources the cluster controller can allocate to each VM on a particular node and reserving resources usually will have more cost for IaaS provider than allocating it in the first place.

Despite all these difficulties, considering all the computing resources in the cluster as a whole, online resource scaling of VMs in IaaS is an obvious benefit both for the cloud provider and the end user, and it completely follows cloud pay-per-use model.

The only way to enable a VM to utilize recourses on the other physical nodes of the cluster is to migrate it to a node with underutilized recourses within the cluster. By leveraging virtual machine migration in the cloud infrastructure, it is possible to regard the cluster as a pool of resources and find an optimum mapping between physical and virtualized resources.

The proposed cloud infrastructure control architecture also addresses how these extended cloud services should be presented to the user and what the trade-offs are between the user and the cloud provider.

A. Parameters

A typical self-managed cluster controller in the cloud infrastructure usually has to consider several parameters to manage resources:

- I, the index set of virtual machines
- K, the index set of physical machines
- C_k, maximum number of virtual cores the hypervisor on physical machine k can allocate to virtual machines
- C_i, virtual CPU cores allocated to virtual machine i
- M_k, maximum amount of memory that the hypervisor on physical machine k is able to allocate to virtual machines
- m_i, memory allocated to virtual machine i
- T_i, migration cost of virtual machine i to a hypervisor on another physical machine. Usually this parameter vastly depends on m_i (Only cluster controllers that support virtual machine migration, use this parameter)

To be able to generalize, we use variable R to indicate all kinds of resources, i.e. CPU virtual cores, memory, network bandwidth or any other resources a hypervisor on a physical machine is able to utilize. Variable r also indicates the corresponding resource on the virtual machine.

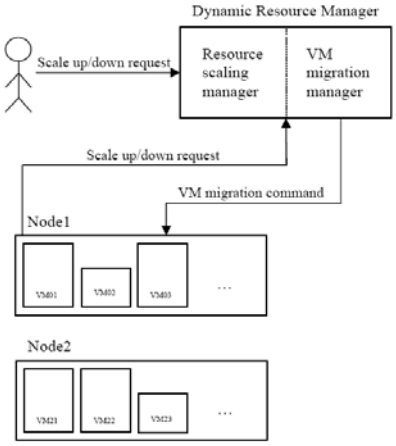

Fig. 2. Dynamic Resource Management in a cluster of the cloud

Our proposed architecture also introduces a couple of new parameters to manage scaling of resources:

- $r_{min,i}$, *minimum* resource requirements for virtual machine *i*
- rmax,i, maximum resource requirements for virtual machine *i*
- $r_{cur,i}$, *current* resource requirements for virtual machine *i*
- $\tau_{min,r,i}$, *minimum* allocation time of resource r to virtual machine *i*
- $\tau_{r,i}$, *allocation* time of resource r to virtual machine *i*
- Usually it is safe to set $r_{max,i}$ to R_k, in which k is the corresponding physical machine for the virtual machine *i* hypervisor, because in absence of other virtual *machines* on that hypervisor, virtual machine *i* could utilize up to R_k amount of physical machine k resources.

B. System Behavior

Each virtual machine in this architecture could be in either of these two modes:

- *resource guaranteed mode*: in this mode, cloud *infrastructure* guarantees scaling of resources up to rmax,i for virtual machine i
- *resource unguaranteed mode*: in this mode, cloud *infrastructure* only guarantees $r_{min,j}$ amount of the resource for virtual *machine* j

The hypervisor on each node of the cluster could have virtual machines in both resource guaranteed and unguaranteed modes. Therefore, to distinguish them, we use $\bar{r}$ to imply the amount of resource in guaranteed mode, while r implies unguaranteed mode. Also, we use I as the index set of virtual machines in resource guaranteed mode while J indicates the index set of virtual machines in unguaranteed mode.

The minimum requirement of a resource on a cluster to make the cloud provider able to guarantee its corresponding service level agreement (SLA) is given by:

$$\sum_{i \in I} \bar{r}_{\max,i} + \sum_{j \in J} r_{\min,j} < \sum_{k \in K} R_k \qquad (1)$$

Although cloud infrastructure does not guarantee more than $r_{min,j}$ for virtual machine j in resource unguaranteed

mode it is beneficial for both cloud provider and the end user to scale up the resource when it is necessary if the cloud has enough unutilized resources. Therefore, a factor α could indicate the ability of the cluster to fulfill the resource scaling requirements of the virtual machines of the cluster in the resource unguaranteed mode.

$$\sum_{k \in K} R_k = \sum_{i \in I} \bar{r}_{\max,i} + \alpha \sum_{j \in J} r_{\max,j} \quad (2)$$

Equation (2) indicates the total amount of available resources in the cluster with respect to α factor. Using a bigger value for α increases the cost of virtual machines in the cluster.

It is possible to use a similar cost factor for virtual machines in the resource guaranteed mode, although choosing values less than 1 could cause infringing the service level agreements.

The cluster controller has to allocate resources to virtual machines in the resource unguaranteed mode, so that (3) will always be feasible:

$$\alpha \sum_{j \in J} r_{\max,j} \geq \sum_{j \in J} r_{cur,j} \quad (3)$$

The cloud provider has to choose the cost factor to determine service level agreement boundaries according to:

$$\frac{\sum_{j \in J} r_{\min,j}}{\sum_{j \in J} r_{\max,j}} < \alpha < 1 \quad (4)$$

It is important to choose the appropriate α for different resources, e.g. the value of α for memory and virtual CPU cores could be completely different.

C. Cloud Control Architecture

After extending the cloud infrastructure cluster controller with resource scaling and VM migration capabilities, the first question that the cloud infrastructure control architecture has to answer is how these features should be accessed. Essentially, according to the cloud model, it is better to implement these features as a couple of cloud services; hence the IaaS capabilities will be extended according to the basic cloud model.

Implementing these features as a set of user-accessible services has its own problems. In a typical implementation of IaaS, end users seldom are aware of the infrastructure and its computing resources. In fact, in a public cloud, because of location transparency, usually the users only have the information about their virtual machine region. Each region in the cloud could contain several clusters and the users could have no information about available resources in neither its physical machines nor other machines in the cluster.

Of course, technically in a private cloud, the cloud provider and the end user could be the same person, but the same location transparency rules apply. Hence, the end user could not be responsible for inner-cluster VM migration without infringing basic IaaS transparency rules.

Online resource scaling usually requires revealing the same information to the user, but fortunately, considering the cluster as a pool of resources, the cloud and its clusters have virtually an infinite amount of resources. Therefore, the cloud could honor user resource scaling requests with full location transparency only if it has a self-managed VM migration unit.

Fig. 2 depicts the transactions between different components of the cloud according to this model. Different clusters of the cloud should have the same components but they could operate independently. As fig. 2 illustrates, a virtual machine could send a resource scale up/down request to the dynamic resource manager in the cloud. Next, the resource scaling manager decides whether to honor this request or not, according to VM mode and available resources in the cluster. The owner of the virtual machine could also send resource scaling requests.

In contrast, VM migration commands could only be generated by cloud infrastructure control architecture. VM migration manger as illustrated in Fig. 2 is responsible for deciding when the VM migration should take place, and where to. The exact mechanism of this module is highly dependant on cloud provider strategies (e.g. lowering the costs or reducing energy consumption) and beyond the scope of this paper, but we will discuss different parameters that VM migration manager should consider before sending the related commands.

In this architecture, dynamic resource manager is a part of the cluster controller that processes requests from the cloud controller, i.e. indirectly from both the users and their virtual machines in the cluster, and works as follows:

Resource scaling manager: This module processes requests indirectly from the users and their virtual machines on the cluster. Resource scale down requests usually will be accepted unconditionally and proceed instantly.

If the virtual machine is in resource guaranteed mode and the demand is not larger that rmax, the scale up requests will also be accepted unconditionally and without regard to τr. If the related node, has enough available unutilized resource, the respective scale up command will be sent to the node instantly. Otherwise, the new configuration will be sent to the VM migration manager for further processing.

If the virtual machine is in resource unguaranteed mode, the request will only be processed if it will not jeopardize future resource guaranteed mode virtual machine requests, although the cloud provider cost policies is not irrelevant. Having enough available unutilized resources on the related node, or the lack thereof, and the migration cost T_i could play a major role in processing requests from virtual machines in resource unguaranteed mode.

VM migration manager: virtual machine migration in a cluster could be necessary in the presence of a new configuration both when creating a new VM instance and when a scale up request from an already running instance is on the schedule. VM migration manager has to find a new mapping between VMs and physical nodes while imposing minimum costs according to the cloud provider policies. The respective parameters are, but not limited to:

- Migration cost of virtual machines in guaranteed mode (T_i for $i \in$ I)
- Migration cost of virtual machines in unguaranteed mode (T_j for $j \in$ J)
- Current allocation of resources on each node ($\sum_{i \in I_N} \bar{r}_{cur,i}$ and $\sum_{j \in J_N} r_{cur,j}$ where I_N and J_N are the sets of virtual machines on the respective node N)

- Maximum allocation of resources on each node ($\sum_{i \in I_N} \bar{r}_{\max,\, i}$ and $\sum_{j \in J_N} r_{\max,\, j}$ where I_N and J_N are the sets of virtual machines on the respective node N)
- Unutilized resources in each node

IV. Experimental Results

A. Implementation

As an open source infrastructure as a service cloud computing platform, Eucalyptus is an exceptional tool to experiment novel ideas in such an environment. Eucalyptus provides a private cloud with almost identical features and interfaces to Amazon public cloud computing platform, EC2, as well as Amazon storage services, S3.

Eucalyptus implements EC2 compatible web services by leveraging Apache Axis2 web service core engine. Expanding these web services has enabled us to experiment with our new proposed architecture in an experimental private cloud. Creating a new instance of a virtual server is a straightforward task in eucalyptus and with the extended version of the web services, migrating and resource scaling of these instances is also quite straightforward.

In the following experiments a set of memory bound dynamic web services is employed on an Apache web server in each instance of the virtual servers. The web server is also expanded with resource monitoring capabilities. Hence the server could ask the cloud for more resources when it is necessary. Eucalyptus supports different virtual machine hypervisors, although the KVM hypervisor is employed in this experiment.

B. Experiment

The resource usage of virtual machines each hosting the exact same set of dynamic web services is analyzed with the same set of requests, but in different resource management schemes. We focus on memory usage as the main resource of the virtual machines and the response time of the web services in these sets of experiments.

Fig. 3 illustrates the response time of different request blocks for two servers in fixed resource scheme. The server with 1024 MBs of memory is able to respond to all requests within seconds but the response time for the other server with 512 MBs of memory is in the range of 0.2 seconds to 276.1 seconds. This huge difference is the result of increasing the number of operating system swap in/out memory pages in the machines with low amounts of memory.

The response time for the same input for servers in resource guaranteed mode and resource unguaranteed mode (r_{min} = 512 MB and r_{max} = 1024 MB) are depicted in Fig. 4. The response times for both cases are noticeably better than the 512 MB fixed memory scheme. In resource guaranteed mode the response time is only 0.4 seconds less than in fixed 1024 MB memory scheme.

Changing dynamic resource manager parameters could change resource usage and service response time dramatically. Hence it is important to tune them up in different conditions. As an example, Fig. 5 illustrates response times of two servers with the same resource management mode but with different allocation times.

Table I sums up the experimental results. Better response time is proportional to better quality of service and lesser resource usage is proportional to lesser virtual server costs.

V. Conclusion

A challenge in the management of cloud infrastructure resources is finding a service abstraction with a balance between user complexity and cloud provider complexity. The proposed resource management scheme has enabled the users to leverage the cloud pay-per-use model further by demanding different amount of resources for each virtual server in an infrastructure as a service environment. The solution enables cloud providers to make better use of their resources, lowering overall costs.

Although our experiment was focused on memory usage as the main source of the costs and quality of service in virtual server environments, it is possible to apply the same scheme to other cloud resources, too. How the proper values for different resource manager parameters have to be chosen remains as an open problem, however.

References

[1] H. A. L. Cavilla, J. A. Whitney, A. M. Scannell, P. Patchin, S. M. Rumble, E. Lara, M. Brudno, and M. Satyanarayanan, "Snowflock: Rapid virtual machine cloning for cloud computing," in *Proceeding of the 4th ACM European Conference on Computer Systems*, ACM, 2009, pp. 1-12.

[2] J. G. Park, J. M. Kim, H. Choi, and Y. C. Woo, "Virtual machine migration in self-managing virtualized server environments," in *Proc. of 11th International Conference on Advanced Communication Technology*, 2009. pp. 2077-2083.

[3] J. Houle, H. A. L. Cavilla, and J. Mulligan, "Towards a ubiquitous cloud computing infrastructure," in *Proc. of 17th IEEE Workshop on Local and Metropolitan Area Networks*, 2010, pp. 1-6.

[4] How to handle 1000′s of concurrent users on a 360MB VPS. [Online]. Available:http://markmaunder.com/2009/12/01/how-to-handle-1000s-of-concurrent-users-on-a-360mb-vps/

[5] J. Peng, X. Zhang, Z. Lei, B. Zhang, W. Zhang, and Q. Li,"Comparison of Several Cloud Computing Platforms," *Information Science and Engineering (ISISE)*, 2009.

Mohammad Razavi is currently pursuing the M.A. degree in the School of Computer Engineering, University of Isfahan, Iran. He has also received his B.A. degree in Software Engineering from University of Isfahan in 2009. His current research interests include Infrastructure as a Service, cloud computing and network security. He is currently a member of International Association of Computer Science and Information Technology (IACSIT).

Kamran Zamanifar Kamran Zamanifar received the B.Sc. and M.Sc degree in Electrical and Electronic Engineering from Faculty of Engineering at University of Tehran (1976-1985), and the Ph.D. degree in Computer Science (Parallel and Distributed Systems) from School of Computer Studies, University of Leeds, England in 1996. Currently, he is associate professor at Engineering Department of University of Isfahan. He is a member of Computer Society of Iran and a senior member of Iranian Association of Electrical and Electronic Engineers. His research interests include parallel and distributed systems, pervasive computing and cloud computing. Prof. Zamanifar has published diverse papers including Time Scheduling and Resource allocation in Computational Systems and Agent-Based Parallel Solution for Job Shop Scheduling Problem Using Genetic Algorithms.