Auto-Scaling Model for Cloud Computing System

Che-Lun Hung[1*], Yu-Chen Hu [2] and Kuan-Ching Li[3]

[1] Dept. of Computer Science & Communication Engineering, Providence University
[2] Dept. of Computer Science & Information Management, Providence University
[3] Dept of Computer Science & Information Engineering, Providence University
{clhung, ychu, kuancli}@pu.edu.tw

**corresponding author*

Abstract

Recently, Cloud Computing introduces some new concepts that entirely change the way applications are built and deployed. Usually, Cloud systems rely on virtualization techniques to allocate computing resources on demand. Thus, scalability is a critical issue to the success of enterprises involved in doing business on the cloud. In this paper, we will describe the novel virtual cluster architecture for dynamic scaling of cloud applications in a virtualized Cloud Computing environment. An auto-scaling algorithm for automated provisioning and balancing of virtual machine resources based on active application sessions will be introduced. Also, the energy cost is considered in the proposed algorithm. Our work has demonstrated the proposed algorithm is capable of handling sudden load requirements, maintaining higher resource utilization and reducing energy cost.

Keywords: *Cloud Computing, Auto-Scaling, Virtualization, Virtual Machine.*

1. Introduction

Recently, Cloud Computing [1, 2, 3] as a new enterprise model has become popular to provide on-demand services to user as needed. Cloud Computing is essentially powerful computing paradigm to deliver services over the network. The model of Cloud Computing has been distinguished into infrastructure-as-a-service (IaaS), Platform-as-a-service (Paas), and software-as-a-service (SaaS) [4].

In Cloud Computing environment, the virtualization technology [5, 6, 7] is the import role to provision the physical resources, such as processors, disk storage, and broadband network. Virtualization refers primarily to platform virtualization, or the abstraction of physical resources for users. In the Could, these physical resources is regarded as a pool of resources, these resources thus can be allocated on demand. Computing at the scale of the Cloud system allows users to access the enormous and elastic resources on-demand. However, a user demand on resources can be various in different time, maintaining sufficient resources to meet peak resource requirements can be costly. On the contrary, if user maintains only minimal computing resources, the resource is not sufficient to handle the peak requirements.

Therefore, dynamic scalability is a critical key point to the success of Cloud Computing environment. Dynamic resizing is a feature allows server to resize the virtual machine to fill the new requirement of resources [7]. When a virtual machine is under-provisioning or over-provisioning, dynamic resizing can utilized to overcome these problem. However, it is not suitable in the Cloud business model as Amazon EC2. In Amazon EC2 [8], a virtual machine is as a basic unit, and user can rent a unit or more units rather than a unit with a special resources. Dynamic resizing cannot overcome the load balancing since it only grows the resources on a specific virtual machine. Trieu C. Chieu *et al.* [9] proposed a useful architecture for dynamic scaling of the web application in a virtualized cloud computing environment. Their architecture includes front-end load-balancer, virtual cluster monitor system and auto-provisioning system, and works efficiently for general web applications in virtual machines. However, a user can rent many virtual machines to perform his applications which are not only web applications. Meanwhile, for the security, virtual machines for a specific user can be regarded as a group, and a user cannot access the resources from other groups.

In this paper, we will present a dynamic-scaling scenario with novel design of applications deployed in virtual machines in a virtual cluster. The rest of the paper is organized as follows. Section 2 illustrates the virtual cluster architecture in a virtualized cloud computing environment. Section 3 describes our auto-scaling algorithm. Finally, section 5 concludes the paper.

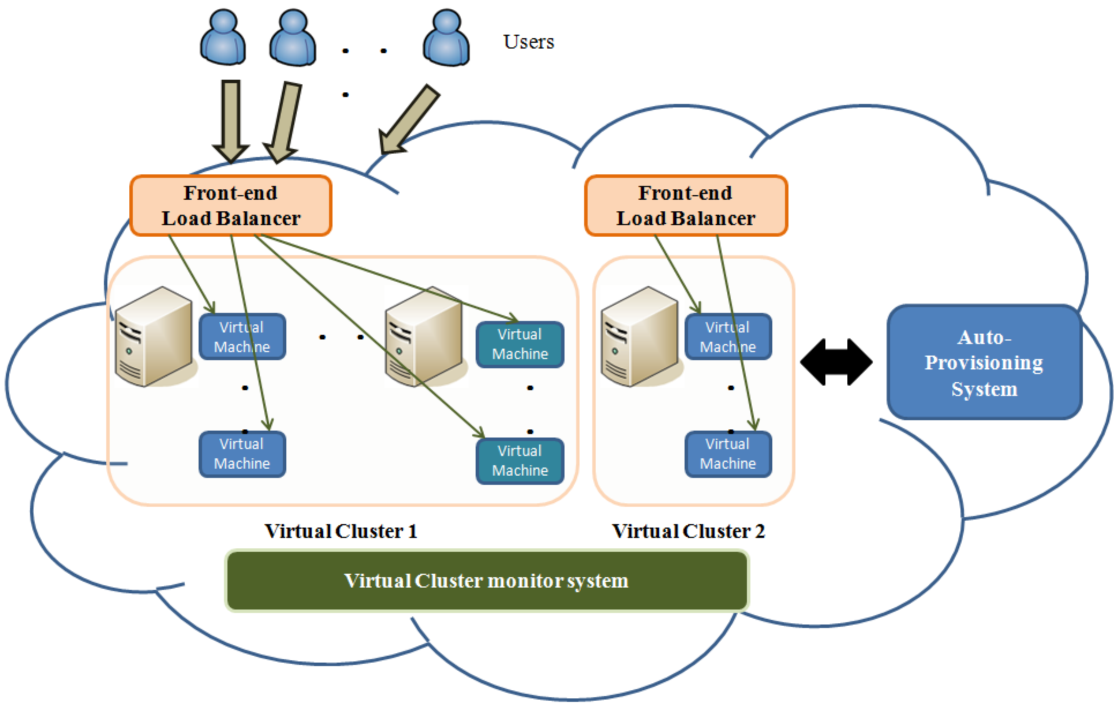

Figure 1. Architecture of the Auto-scaling in a Cloud

2. Architecture Design

We consider two scenarios about web service and parallel processing application in the Cloud Computing environment. A web service should be available at anytime and should provide the fastest response time regardless of the number of users served. Therefore, a Cloud service system should scale the service dynamically; extending and shrinking the number of web servers and web service components for large requirement and small requirement. Cloud Computing also allows users to access supercomputer-level computing power by distributing tasks into large amount of computing nodes (virtual machines). Users may not know the exact number of computing nodes should be utilized to efficiently perform their jobs. Thus, the under-provisioning and over-provisioning happen often. A Cloud computing system should scale the computing nodes according to the workload. A scalable architecture that effectively handles these two scenarios is illustrated in Figure 1. This architecture includes three main components: Front-end load balancer, Virtual cluster monitor system and Auto-provisioning system with an auto-scaling algorithm. Front-end load balancer can balance the requests between different virtual machines that perform the same application in a virtual cluster. Virtual cluster monitor system collects resource usages of all VMs for each virtual cluster that are running on Cloud. Auto-provisioning system can horizontally expand/shrink the number of VMs according to workload of a virtual cluster where VMs belong. If the application running on the virtual cluster consumes most of the resources, auto-scaling can create a new VM that perform the same application and front-end load balancer then balances the requests among VMs in the same virtual cluster.

3. Load Balancing

In our architecture, Front-end load balancer is utilized to balancing the web application load. Apache HTTP Load-Balancer is utilized as Front-end load balancer to allow incoming HTTP request to be routed into web servers that perform the web application. Since the Apache HTTP Load-Balancer configuration can be updated dynamically, this allows a Cloud system to automatically and dynamically add new web servers for a virtual cluster. The additional web servers enable the virtual cluster to scale and thus provide better response time for incoming HTTP requests.

```
Auto-Scaling Algorithm1
Input:   $n$: number of Clusters
        $VC$: an Virtual Cluster consists of VMs that run the same computational system
        $VM_{ns}$: number of the active sessions in a virtual machine
        $S_{iMax}$: maximum sessions for a virtual machine of $i$-th Cluster
        $S_{upper\_bound}$: session upper-threshold
        $S_{low\_bound}$: session low-threshold
        $E_{below}$: a virtual machine set records virtual machines that exceed the session upper-threshold
Output: Front Load-Balancer Set FLB

1.  For $i$ = 1 to $n$
2.     For each VM $\in VC_i$
3.        If ($VM_{ns}/S_{iMax} \geq S_{upper\_bound}$) then
4.          $e = e + 1$
5.        If ($VM_{ns}/S_{iMax} \leq S_{low\_bound}$) Then
6.          $b = b + 1$
7.          Record VM to $E_{below}$
8.     If ($e$ == | $VC_i$|) then
9.        Provision and start a new VM that runs the same system as $VC_i$
10.       Add new VM to FLB //Front Load-Balancer Set
11.    If ($b \geq 2$) then
12.       For each VM in $E_{below}$
13.          If ($VM_{ns}$ == 0) then
14.            Remove VM from FLB //Front Load-Balancer Set
15.            Destroy VM
16.    Empty $E_{below}$
```

Figure 2. Auto-Scaling Algorithm for Web Applications

4. Auto-Scaling Algorithm

For web application, Virtual cluster monitor system can detect whether the number of active HTTP sessions are over the threshold in a virtual cluster. For distributed computing task, Virtual cluster monitor system is able to detect whether the number of virtual machine are over the threshold of use of physical resources in a virtual cluster. As shown in Figure 1, the auto-scaling algorithm is implemented in Auto-provisioning system, and Virtual cluster monitor system is used to control and trigger the scale-up and scale-down in Auto-provisioning system on the number of virtual machine instances based on the statistics of the scaling indicator.

Installation of a software appliance on a virtual machine creates a virtual appliance. Virtual appliance image is a set of virtual appliances on a virtual machine. A virtual appliance image includes guest OS and applications, and it can eliminate the installation, configuration and maintenance costs associated with running complex stacks of software. Therefore, to simplify provisioning process, a virtual machine appliance image template that includes the web application and its corresponding web server is stored in the image repository of the Cloud system. The new virtual machines of web servers and web applications can be rapidly created and provisioned to the virtual cluster using the corresponding appliance image template.

For web service application, network bandwidth and number of sessions are most important index for the service level agreement (SLA) and quality of service (QoS). Figure 2 illustrates our auto-scaling algorithm for web service application. The algorithm first determines the current VMs with network bandwidth and active sessions above or below given threshold numbers, respectively. If all VMs have network bandwidth and active sessions above the given upper threshold, a new VM will be provisioned, started, and then added to the front-end load-balancer, respectively. If there are VMs with network bandwidth and active sessions below a given lower threshold and with at least one VM that has no network traffic or active sessions, the idle VM will be removed from the front-end load-balancer and be terminated from the system.

For distributed computing tasks, physical computing resources such as the uses of CPU and memory are most important indices for evolution of workload of a virtual cluster. Figure 3 illustrates our auto-scaling algorithm for distributed computing tasks. The algorithm first determines the current VMs with the use of physical resources above or below given threshold numbers. If the uses of resources of all VMs are above the given upper threshold, a new VM will be created, provisioned, started, and then perform the same computing tasks in the virtual cluster. If the uses of resources of some VMs are below a given lower threshold and with at least one VM that has no computing job, the idle VM will be terminated from the virtual cluster.

5. Conclusion

In this paper, we have presented two scaling scenarios to address the automatic scalability of web applications and distributed computing jobs in a virtual cluster on the virtualized Cloud Computing environment. The proposed Cloud Computing architecture is constructed with a Front-end load balancer, a Virtual cluster monitor system and an Auto-provisioning system. The Front-end load balancer is utilized to route and balance user requests to Cloud services deployed in a virtual cluster. The Virtual cluster monitor system is used to collect the use of physical resources of each virtual machine in a virtual cluster. The Auto-provisioning system is utilized to dynamically provision the virtual machines based on the number of the active sessions or the use of the resources in a virtual cluster. In addition, the idle virtual machines are destroyed, and then the resources are able to be released. In the other hand, the energy cost can be reduced by remove the idle virtual machines. Our work has demonstrated the proposed algorithm is capable of handling sudden load requirements, maintaining higher resource utilization and reducing energy cost. The auto-scaling mechanism of the Cloud system is the essential element in enhancing the resource utilization, thus reducing the infrastructure and management costs.

```
Auto-Scaling Algorithm2
Input:   $n$: number of Clusters
        $VC$: an Virtual Cluster consists of VMs that run the same
computational system
        $VM_R$: The use of resources in a virtual machine
        $R_{upper\_bound}$: The upper-threshold of use of physical resources
        $R_{low\_bound}$: The low-threshold of use of physical resources
        $V_{below}$: a virtual machine set records virtual machines that

1.  For $i$ = 1 to $n$
2.     For each VM $\in$ $VC_i$
3.        If ($VM_R \geq R_{upper\_bound}$) then
4.          $e = e + 1$
5.        If ($VM_R \leq R_{low\_bound}$) Then
6.          $b = b + 1$
7.          Record VM to $V_{below}$
8.     If ($e$ == | $VC_i$|) then
9.        Provision and start a new VM that runs the same computing
    tasks as $VC_i$
10.    If ($b \geq 2$) then
11.       For each VM in $V_{below}$
12.          If (VM is idle) then
13.            Destroy VM
14.    Empty $V_{below}$
```

Figure 3. Auto-Scaling Algorithm for Distributed Computing Tasks

Acknowledgements

This research was partially supported by the National Science Council under the Grants NSC-100-2221-E-126-006 and NSC-99-2632-E-126-001-MY3.

References

[1] E. Knorr and G. Gruman, InfoWorld, **(2010)**.
[2] R. Buyya, Y. S. Chee and V. Srikumar, Dept. Computer Science and Software Engineering, University of Melbourne, Australia, **(2008)**.
[3] D. Chappel, David Chappell & Associates, **(2008)**.
[4] R. Prodan and S. Ostermann, Proceedings of the 10th IEEE/ACM international conference on Grid Computing, **(2009)** October 13-15; Banff, Canada.
[5] VMware Inc., http://www.vmware.com/products/vi/esx/.
[6] Xen, http//www.xen.org.
[7] Kernel-based Virtual Machine (KVM), http://www.linux-kvm.org.
[8] Amazon EC2, http://aws.amazon.com/ec2/.
[9] T. C. Chieu, A. Mohindra, A. A. Karve and A. Segal, Proceedings of the IEEE international Conference on e-Business Engineering, **(2009)** October 21-23; Macau, China.