# High Availability on Cloud with HA-OSCAR

Thanadech Thanakornworakij[1], Rajan Sharma[1], Blaine Scroggs[1], Chokchai (Box) Leangsuksun[1], Zeno Dixon Greenwood[1], Pierre Riteau[2,3], and Christine Morin[3]

[1] College of Engineering & Science, Louisiana Tech University, Ruston, LA 71270, USA
{tth010,rsh018,bas031,box}@latech.edu, greenw@phys.latech.edu
[2] Université de Rennes 1, IRISA
[3] INRIA Rennes – Bretagne Atlantique
Pierre.Riteau@irisa.fr, Christine.Morin@inria.fr

**Abstract.** Cloud computing provides virtual resources so that end-users or organizations can buy computing power as a public utility. Cloud service providers however must strive to ensure good QoS by offering highly available services with dynamically scalable resources. HA-OSCAR is an open source High Availability (HA) solution for HPC/cloud that offers component redundancy, failure detection, and automatic fail-over. In this paper, we describe HA-OSCAR as a cloud platform and analyze system availability of two potential cloud computing systems, OSCAR-V cluster and HA-OSCAR-V. We also explore our case study to improve Nimbus, a popular cloud IaaS toolkit. The results show that the system that deploys HA-OSCAR has a significantly higher degree of availability.

**Keywords:** HA-OSCAR, High Availability, OSCAR-V.

## 1 Introduction

Cloud computing refers to a service-oriented paradigm where service providers offer the computing resources such as hardware, software, storage and platforms as services according to the demands of the user. The benefit of cloud computing is to increase utilization of available computing resources and reduction of burden and responsibilities of end-users by renting resources, and thus, increase economic efficiency [1]. Cloud computing collects computing resources and manages them automatically through dynamic provisioning and often virtualized resources. The user or client companies do not deal with software and hardware administrative issues, as they can buy these virtual resources through the cloud service providers depending on their needs [2]. The focus of cloud computing is to provide easy, secure, fast, convenient and inexpensive net computing and data storage service centered on the Internet. This however transfers such responsibilities to service providers in order to ensure QoS.

HA systems are increasingly vital and important due to their ability to sustain critical services to users. Also HA services are very important in clouds because companies and users depend on these cloud providers for their critical data. In order for the cloud computing to be effective in business, scientific research etc., high availability is a must. Thus, we foresee that it is critically important that we enable cloud infrastructure with HA.

The HA-OSCAR [7] project originally started from OSCAR (Open Source Cluster Application Resource) project. OSCAR is a cluster software stack that provides a high performance computing runtime stack and tools for cluster computing [5]. The main goal of the HA-OSCAR project was to leverage existing OSCAR technology, so the HA-OSCAR project was formed to provide high-availability capabilities in OSCAR clusters. HA-OSCAR then introduces several enhancements and new features to OSCAR mainly in areas of availability, scalability and security [5],[11]. Initially HAOSCAR [8] only supported OSCAR clusters, however, the current version supports most Linux-based IT infrastructures, and not just OSCAR clusters. Thus, HA-OSCAR is a capable cloud platform that enables not only scalability aspect via its cluster computing capability with OSCAR, or the like, but also HA solutions.

In this paper we introduce a new system, HA-OSCAR 2.0 [14] (an Open Solution HA-enabling framework for mission critical systems), capable of enhancing HA in cloud platforms by adopting component redundancy to eliminate single-point-of-failures. Thus, system critical resources are replicated so that failures of any resources will not take down the entire system, thereby making cloud infrastructure highly available, especially at the head node. Some of the new and improved features this version incorporates are self-healing mechanisms, failure detection, automatic synchronization, and, fail-over and fail-back functionality [7], [14].

## 2 Related Work

OSCAR –V [6] provides an enhanced set of tools and packages for the creation, deployment and the management of virtual machines and host operating system within a physical cluster. Virtualization in OSCAR clusters is needed to decouple the operating system, customize the execution environment, and provide computing based on the need of the user. Increased HA thus makes it perfectly compatible with HPC cloud computation. OSCAR-V uses Xen as the virtualization solution and provides V2M virtual machine management of physical clusters.

Another interesting and highly scalable cluster solution in cloud computing is Rocks+ [9]. It can be used for running a public cloud or setting up an internal private cloud. Rocks+ is based upon the well-known software Rocks. Rocks contain all the necessary software components required to easily build or maintain a cluster or cloud. Rocks+ can manage an entire data center running all the computational resources and services necessary to operate a cloud infrastructure with a single management point. Rocks+, with Rolls pre-owned software, allows users to build web servers, database servers, and compute servers in the cloud. Rocks+ also provides a framework for user-specific needs on clouds. Rocks+ provides CPU and GPU cluster management in less time and with lower costs through software automation. [10].

Availability is a crucial factor in cloud computing services. Many studies attempt to balance HA, based on cloud system performance and cost. Jung [3] studied a replication technique to guarantee HA while maximizing performance on a certain number of resources. In Ref [3], replication of software components is used to provide HA. In case of hardware failure, they used component redundancy and regenerated the software components into the remaining resources to achieve HA and optimize performance, based on a queuing model with different "mean time between failure" (MTBF) and "mean time to repair" (MTTR).

## 3 HA-OSCAR 2.0

HA-OSCAR 2.0 is an open source framework that provides HA for mission critical applications and ease of system provisioning and administration. The main goal of the new HA-OSCAR Project is to provide an open solution, with improved flexibility, which seeks to combine the power of HA and High Performance Computing (HPC) solutions. It is capable of enhancing HA for potential cloud computing infrastructure such as web services, and HPC clouds by providing the much-needed redundancy for mission critical applications. To achieve HA, HA-OSCAR 2.0 uses HATCI (High Availability Tools Configuration and Installation). HATCI is composed of three components: Node Redundancy, Service Redundancy and Data Replication Services. The installation process requires just a few steps with minimum user input. HA-OSCAR 2.0 incorporates a feature to clone the system in the installation step to make the data and software stacks consistent on the standby head node or a cloud gateway. If the primary component fails, the cloned node takes over the responsibilities. HA-OSCAR also features monitoring services with a flexible event-driven rule-based system. Moreover, it provides data synchronization between the primary and secondary system. All of these features are enabled in the installation process.

### *3.1* HA-OSCAR Software Architecture

HA-OSCAR 2.0 combines many existing technology packages together to provide HA solution. HA-OSCAR software architecture has three components. The first component is IP monitoring using Heartbeat. Heartbeat is a service designed to detect failure of the physical components such as network and IP services. Heartbeat uses virtual IP address to make the fail back mechanism possible. When the primary node is not in a healthy mode, Heartbeat handles IP fail-over and fail-back mechanisms. The second component is the service monitor, MONIT. MONIT is a small and lightweight service used to monitor the health of important services to make those services highly available. MONIT will attempt to restart the failing services for 4 times by default and is tunable. If the services cannot be successfully restarted, MONIT will send the message to Heartbeat to trigger fail-over. This can be customized to meet the needs of the user. The third component, data synchronization, is provided by a service called HA-filemon. HA-filemon is a daemon that monitors changes made in the given directory trees and provides replication service via rsync accordingly. This is a new module that was not available in earlier versions of HA-OSCAR. During a fail-back event, data will synchronize from the standby to the primary server. This backwards synchronization occurs to propagate changes made to the secondary server while it is the head node. By default, HA-filemon will invoke rsync 2 minutes after it detects the first change in files, to allow groups of changes to transmit together. Users can change this time according to the need of their applications. System Imager is used for cloning the primary node during the installation process. It creates a standby head node image from the primary server. Finally, HA-OSCAR 2.0 supports virtual machine management via integration with OSCAR-V.

The incorporation of all the above services endows HA-OSCAR 2.0 with the ability to provide true HA and high performance for cloud users, for whom critical services must be guaranteed. Thus HA-OSCAR 2.0 is a potentially viable open source solution for achieving HA in cloud computing.

A mission-critical web service is an example where HA-OSCAR 2.0 can be applied to a cloud. Hardware or software failure, and routine maintenance are potential factors affecting services unavailability. Installing HA-OSCAR to provide component redundancy can alleviate this problem. In the installation process of HA-OSCAR 2.0 for web services, a clone of the primary web server is made that acts as a standby server, and maintains data synchronization between primary and standby server for data consistency. The primary web server receives requests from clients and serves the requests directly or reroutes them to a web farm via LVS [15] or the like. When failure occurs on the primary web server, the standby web server will take over as primary web server. It will automatically be configured with the same IP address as the primary web server so that all the requests are redirected to the standby server in the same cloud advertised address, making the web service highly available. When the primary web server is available again, and the repair is completed, by default this server will become the standby server. If users need the fixed server to be the primary server, they have to run the fail-back script to make the fixed server work as primary web server.

## 4 System Architecture

In this section, we first examine the OSCAR-V architecture as a potential HPC cloud and its anatomy, in order to identify single-point-of-failure components. This will provide an opportunity to introduce system level redundancy that will produce a HA initiated improvement over the existing OSCAR-V cluster framework. However, only a brief description of the proposed architecture is entailed here. Additional HA-OSCAR details may be found in [14].

### 4.1 OSCAR-V Cluster Architecture

Figure 1 shows the architecture of an OSCAR-V cluster system. Each individual machine within a cluster is referred to as a node. There are two types of nodes: head nodes and compute nodes. The head node provides service requests and routes appropriate tasks to compute nodes. Compute nodes are primarily dedicated to computation. The present OSCAR-V cluster architecture consists of a single server node and a number of client nodes, where all the client nodes can be virtualized by Xen virtualization solution.

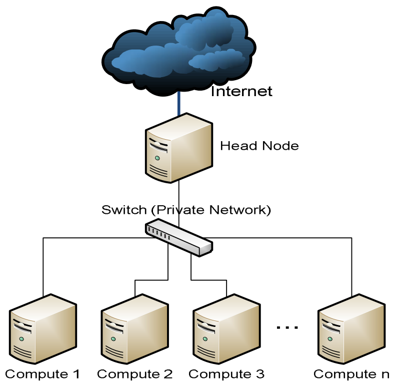

**Fig. 1.** OSCAR-V Architecture

### 4.2 HA-OSCAR-V Cluster Architecture

As a proof-of-concept for HPC clouds, we employ OSCAR-V as a solution enabling users to deploy virtualized clusters. In order to support HA requirements, clustered systems must provide ways to eliminate single-point-of-failures. Hardware and network redundancy are common techniques utilized for improving the availability of computer systems. To achieve our cloud platform, we must first provide a redundant cluster head node. The dual master head nodes will replicate the data and configurations by using Rsync. In the event of a head node outage, all functionalities provided by that primary head node will fail-over to the second redundant head node. An additional HA functionality supported in HA-OSCAR is that of providing a high-availability network via redundant Ethernet ports on every machine in addition to duplicate switching fabrics (network switches, cables, etc.) for the entire network configuration.

Figure 2 shows the typical HPC cloud architecture enabled by HA-OSCAR solution. Each of the redundant server nodes is automatically installed and configured as well as connected to an external network by two or more different links. These redundant links will keep the system connected to the external environment if one network, each server node is also connected to one or more switches and each client node is connected to both switches providing optional two redundant switching fabrics.

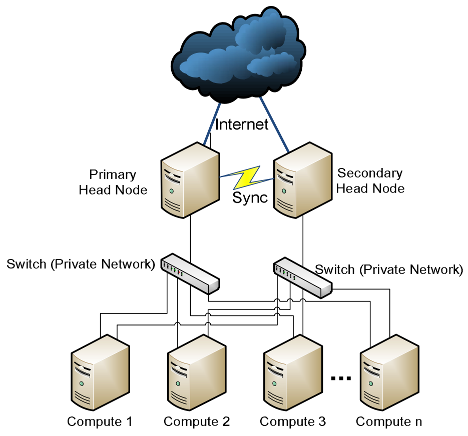

**Fig. 2.** A typical HA-OSCAR-V cluster system

### 4.3 Case Study: HA-OSCAR-Enabled Nimbus

The Nimbus IaaS toolkit [16] is software that allows deploying an Infrastructure-as-a-Service cloud, similar to what the Amazon EC2 platform**.** offers. Users interact with a central service in order to request virtual machines (individual ones or clusters), get their status, terminate them, etc. Additionally, they interact with a storage repository to upload and download virtual machine images. This repository is named Cumulus and implements the Amazon S3 API.Hence, fault tolerance capabilities for these two components are vital requirements for ensuring high availability of a Nimbus cloud crucial services. In case of a failure, users are not capable of monitoring and managing their virtual machines or modifying their statuses.

Using HA-OSCAR 3 main services, namely replication, monitoring and synchronization, it would be possible to provide standby nimbus services, synchronize Nimbus configuration files, internal databases (where information about users and VMs is persisted), and repository content (such as newly uploaded VM images). When a failure occurs, HA-OSCAR would failover to the standby node in order to ensure the critical service availability. For now, we consider only high availability of the Nimbus service and repository, providing transparent high availability of the VMs running in the cloud in a more complex problem, as shown by previous research [17].

## 5 System Model

In this section, we evaluate availability improvement when deploying HA-OSCAR on a regular cluster cloud. We first define the state space reliability model [13] for system availability evaluation. In our case, let's consider a OSCAR-V and HA-OSCAR integrated solution. Our analysis focuses on servers and switches that dominate cluster availability since there are many more compute nodes in the HPC cloud that will not suffer from single point of failure. We made several assumptions for the state-space as follows:

- Time to failure for both virtualization system servers and switches is exponentially distributed, with the parameters λv for the servers and λw for the switches, respectively. We consider the failure of the Virtualization system server that has virtualization and the physical server as one component.
- Failed components can be repaired.
- Times to repair for a server and switches are exponentially distributed with parameters μ and β.
- When the system is down, no further failure can take place. Hence, for the OSCAR-V cluster, when the server is down, no further failure can take place on the switch. Similarly, when the switch is down, no further failure can take place on the server. For HA-OSCAR-V clusters, when both servers are down, no further failure can take place on the switches. Similarly, when both switches are down, no further failure can take place on the HA-OSCAR-V cluster.

## 5.1 OSCAR-V Cluster System Model

Figure 3 shows the Continuous Time Markov Chain (CTMC) model [13] corresponding to the OSCAR-V cluster system. In state 1, both server nodes and switches are well functioning. The transition to state 2 occurs if a switch node has a failure and the transition from state 1 to state 3 occurs when a server has a failure. The system will be available for service in state 1, and will be unavailable for state 2 and state 3. The system goes from state 1 to state 2 when switch failure occurs at rate λw , and from state 1 to state 3 when server failure occurs at rate λv . After switch recovery at rate β, the system is back in state 1 from state 2. Moreover, after server recovery at rate μ, the system is back in state 1 from state 3. To have HA, we must try to keep the system in the state 1 as long as possible.

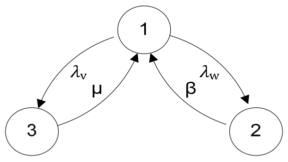

**Fig. 3.** CTMC diagram for OSCAR-V cluster system

## 5.2 HA-OSCAR-V Cluster System Model

Figure 4 shows the CTMC [12], [13] model corresponding to HA-OSCAR-V cluster system. Table 1 shows the states, number of operating components, and their corresponding system status. The system is available for service in states 1, 2, 4 and 5, and is unavailable in states 3, 6, 7 and 8. The system goes from one state to another at the rates showed in the arrow lines in Figure 4.

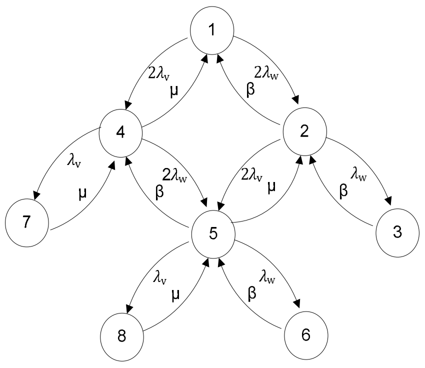

**Fig. 4.** CTMC diagram for HA-OSCAR-V cluster system

**Table 1.** System status

| State Number | Number of Operating Servers | Number of Operating Switches | System Status |
|---|---|---|---|
| 1 | 2 | 2 | Up |
| 2 | 2 | 1 | Up |
| 3 | 2 | 0 | Down |
| 4 | 1 | 2 | Up |
| 5 | 1 | 1 | Up |
| 6 | 1 | 0 | Down |
| 7 | 0 | 2 | Down |
| 8 | 0 | 1 | Down |

## 6 Availability Analysis

Let $\pi_i$ be the steady-state probability of state *i* of the CTMC. They will satisfy the following equations:

$$\sum_{i\in\Omega} \pi_i = 1 \quad \text{and} \quad \pi Q = 0 ,$$

where *Q* is the infinitesimal generator matrix [13].

Let *U* be the set of up states, the availability of the system *A* is

$$A = \sum_{k\in U} \pi_k .$$

### 6.1 OSCAR-V Cluster System Analysis

We calculate the steady-state probabilities by balance equations. For this model, state 1 is the only state that available for the service. We have the steady-state availability

$$A = \pi_1 = \frac{1}{E} , \tag{1}$$

where

$$E = 1 + \frac{\lambda_w}{\beta} + \frac{\lambda_v}{\mu}.$$

### 6.2 HA-OSCAR-V Cluster System Analysis

Considering the operating states, which is the system still responding to the request for the service; we compute availability from the steady-state probabilities by balance equations. We then have the steady-state availability.

$$A = \pi_1 + \pi_2 + \pi_4 + \pi_5$$
$$= \frac{1 + \frac{2\lambda_w}{\beta} + \frac{2\lambda_v}{\mu} + \frac{4\lambda_w\lambda_v}{\beta\mu}}{E}, \qquad (2)$$

where

$$E = 1 + \frac{2\lambda_w}{\beta} + \frac{2\lambda_v}{\mu} + \frac{2\lambda_w^2}{\beta^2} + \frac{2\lambda_v^2}{\mu^2} + \frac{4\lambda_w\lambda_v}{\beta\mu} + \frac{4\lambda_w^2\lambda_v}{\mu\beta^2} + \frac{4\lambda_w\lambda_v^2}{\mu^2\beta}.$$

### 6.3 HA Comparison

We assume that λv = *0.001hr-1,* λw = *0.0005 hr-1,* μ = *0.5 hr-1, and* β = *1.0 hr-1.* With equations 1, 2, and 3 [13] , we can calculate the availability of the system. The availability for OSCAR-V server cluster is 0.996, and the availability for the HA-OSCAR-V cluster system is 0.99999. The downtime of the two systems in a year is 39.2 hours and 4.45 minutes, respectively. Typically a HA system is one that has a downtime that does not exceed "five-nines" or 99.999%.

## 7 Conclusions and Future Work

HA is an important factor for Cloud service providers to ensure QoS and to meet SLA. HA-OSCAR aims to improve HA of any Linux-based cloud computing platform. The results of our analysis and comparison of the experimental and theoretical HA of OSCAR-V and HA-OSCAR-V cluster systems show that the availability of HA-OSCAR-V cluster systems is significantly higher than that of OSCAR-V cluster systems. HA-OSCAR 2.0 now supports any cluster or enterprise system to improve HA, not only for OSCAR or OSCAR-V but also any Linux system based on Debian distributions. We plan to explore additional ways to extend the availability, robustness and reliability of a HA-OSCAR system to other cloud environment including storage clouds.

## References

1. Zhang, S., Zhang, S., Chen, X., Huo, X.: Cloud Computing Research and Development Trend. In: Second International Conference on Future Networks, ICFN 2010, January 22-24, pp. 93–97 (2010)

2. Zhang, S., Zhang, S., Chen, X., Wu, S.: Analysis and Research of Cloud Computing System Instance. In: 2010 Second International Conference on Future Networks, ICFN 2010, pp. 88–92 (2010)
3. Jung, G., Joshi, K.R., Hiltunen, M.A.: Performance and Availability Aware Regeneration for Cloud Based Multitier Application. In: Dependable Systems and Networks (DSN), pp. 497–506 (2010)
4. Cully, B., Lefebvre, G., Meyer, D., Feeley, M., Hutchinson, N., Warfield, A.: Remus: High Availability via Asynchronous Virtual Machine Replication. In: 5th USENIX Symposium on Networked Systems Design and Implementation (2008)
5. Brim, M.J., Mattson, T.G., Scott, S.L.: OSCAR: Open Source Cluster Application Resources. In: Ottawa Linux Symposium 2001, Ottawa, Canada (2001)
6. OSCAR-V, `http://www.csm.ornl.gov/srt/oscarv/`
7. Leangsuksun, C., Liu, T., Scott, S.L., Libby, R., Haddad, I., et al.: HA-OSCAR Release 1.0: Unleashing HABeowulf. In: International Symposium on High Performance Computing Systems (HPCS), Canada (May 2004)
8. Haddad, I., Leangsuksun, C., Scott, S.L.: HA-OSCAR: the birth of highly available OSCAR. Linux J. 2003(115), 1 (2003)
9. Rock+, `http://www.clustercorp.com/`
10. `http://www.hpcwire.com/offthewire/Clustercorp-Brings-Rocks-to-the-Cloud-108706864.html`
11. Leangsuksun, C.B., Shen, L., Liu, T., Scott, S.L.: Achieving HA and performance computing with an HA-OSCAR cluster. Future Generation Computing Syst. 21(4), 597–606 (2005)
12. Leangsuksun, C., Shen, L., Song, H., Scott, S.L., Haddad, I.: The Modeling and Dependability Analysis of High Availability OSCAR Cluster. In: The 17th Annual International Symposium on High Performance Computing Systems and Applications, Quebec, Canada, pp. 11–14 (May 2003)
13. Trivedi, K.S.: Probability and Statistics with Reliability, Queuing, and Computer Science Applications. John Wiley and Sons, New York (2001)
14. HA-OSCAR 2.0, `http://hpci.latech.edu/blog/?page_id=45`
15. Linux Virtual Server (LVS), `http://www.linuxvirtualserver.org/`
16. Nimbus, `http://www.nimbusproject.org`
17. Nicholas, B., Papaioannou, T.G., Aberer, K.: An Economic Approach for Scalable and Highly-Available Distributed Applications. In: IEEE International Conference on Cloud Computing (2010)