# Highly Reliable Linux HPC Clusters: Self-Awareness Approach

Chokchai Leangsuksun[1,*], Tong Liu[2], Yudan Liu[1,*], Stephen L. Scott[3,**], Richard Libby[4], and Ibrahim Haddad[5]

[1] Computer Science Department, Louisiana Tech University
[2] Enterprise Platforms Group, Dell Corp.,
[3] Oak Ridge National Laboratory
[4] Intel Corporation
[5] Ericsson Research
{box, yli010}@latech.edu, Tong_Liu@dell.com, scottsl@ornl.gov, rml@hpc.intel.com, ibrahim.haddad@ericsson.com

**Abstract.** Current solutions for fault-tolerance in HPC systems focus on dealing with the result of a failure. However, most are unable to handle runtime system configuration changes caused by transient failures and require a complete restart of the entire machine. The recently released HA-OSCAR software stack is one such effort making inroads here. This paper discusses detailed solutions for the high-availability and serviceability enhancement of clusters by HA-OSCAR via multi-head-node failover and a service level fault tolerance mechanism. Our solution employs self-configuration and introduces Adaptive Self Healing (ASH) techniques. HA-OSCAR availability improvement analysis was also conducted with various sensitivity factors. Finally, the paper also entails the details of the system layering strategy, dependability modeling, and analysis of an actual experimental system by a Petri net-based model, Stochastic Reword Net (SRN).

## 1 Introduction

One of the challenges in a clustered environment is to keep system failure to a minimum and to provide the highest possible level of system availability. If not resolved in a timely fashion, such failures can often result in service unavailability/outage that may impact businesses, productivity, national security, and our everyday lives. High-Availability (HA) computing strives to avoid the problems of unexpected failures through active redundancy and preemptive measures. Systems that have the ability to hot-swap hardware components can be kept alive by an OS runtime environment that understands the concept of dynamic system configuration. Furthermore, multiple

[*] Research supported by Center for Entrepreneurship and Information Technology, Louisiana Tech University.

[**] Research supported by U. S. Department of Energy grant, under contract No. DE-AC05-00OR22725 with UT-Battelle, LLC.

head-nodes (sometimes called service nodes) can be used to distribute workload while consistently replicating OS runtime and critical services, as well as configuration information for fault-tolerance. As long as one or more head-nodes survive, the system can be kept consistent, accessible and manageable.

## 2 HA-OSCAR Architecture

A typical Beowulf cluster consists of two node types: a head node server and multiple identical client nodes. A server or head node is responsible for serving user requests and distributing them to clients via scheduling/queuing software. Clients or compute nodes are normally dedicated to computation [4]. However, this single head-node architecture is a single-point-of-failure-prone of which the head-node outage can render the entire cluster unusable.

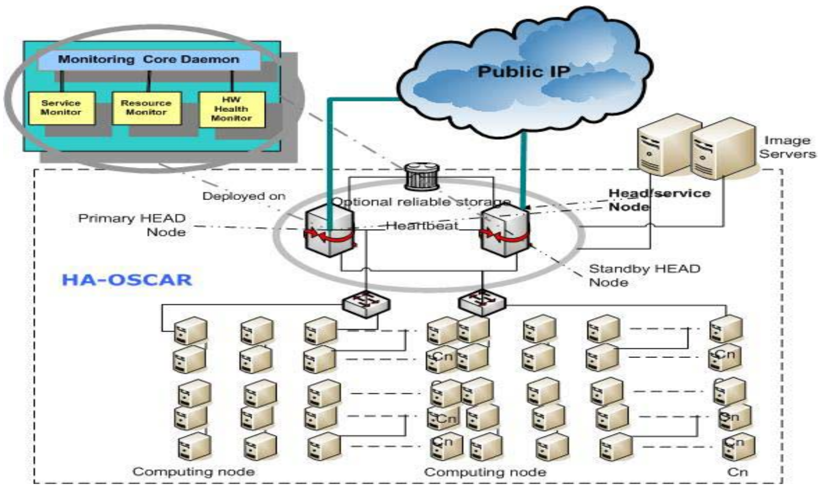


**Fig. 1.** HA-OSCAR architecture

There are various techniques to implement cluster architecture with high-availability. These techniques include active/active, active/hot-standby, and active/cold standby. In the active/active, both head nodes simultaneously provide services to external requests and once one head is down, the other will take over total control. Whereas, a hot standby head node monitors system health and only takes over control when there is an outage at the primary head node. The cold standby architecture is very similar to the hot standby, except that the backup head is activated from a cold start.

Our key effort focused on "simplicity" by supporting a self-cloning of cluster master node (redundancy and automatic failover). Although, the aforementioned failover concepts are not new, HA-OSCAR simple installation, combined HA and HPC architecture are unique and its 1.0 beta release is the first known field-grade HA-Beowulf cluster release.

## 3 HA-OSCAR Serviceability Core

Our existing HA-OSCAR and OSCAR installation and deployment mechanism employs Self-build- a self-configuration approach through an open source OS image capture and configuration tool, SystemImager, to clone and build images for both compute and standby head nodes. Cloned images can be stored in a separate image servers (see Figure 1) which facilitate upgrade and improve reliability with potential rollback and disaster recovery.

### 3.1 Head-Node Cloning

As stated in Section 3, our approach of removing the head node single-point-of-failure is to provide a hot-standby for the active head node. The hot-standby head node is a mirror of the active node and will process user requests when the active head node fails. To simplify the construction of this environment, we begin with a standard OSCAR cluster build on the active node and then "clone" or duplicate that to the hot-standby head node. As shown in Fig. 3, the network configuration differs between the active and hot-standby node by the public IP address, thus they are not exact duplicates of one another. Presently, hardware must be identical between the active and standby machine. Once cloned, both head nodes will contain the Linux operating system and all the necessary components for an OSCAR cluster [4] including: C3, LAM/MPI, LUI, Maui PBS Scheduler, MPICH, OpenSSH, OpenSSL, PBS, PVM, and System Installation Suite (SIS).

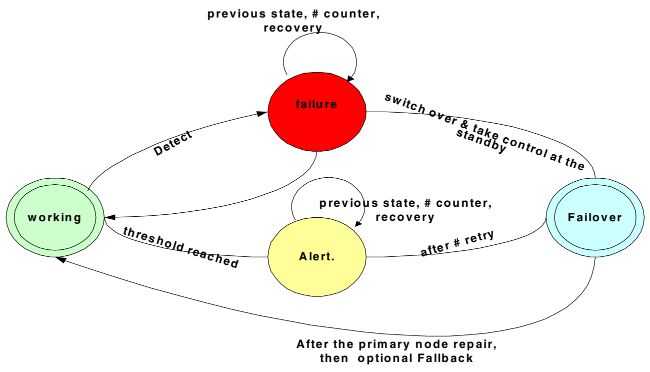

previous state, # counter, recovery
failure
switch over & take control at the standby
Detect
working
previous state, # counter, recovery
Failover
threshold reached
Alert.
after # retry
After the primary node repair, then optional Fallback

**Fig. 2.** Adaptive Self-Healing State Diagram

### 3.2 Adaptive Self-Healing (ASH) Technique

HA-OSCAR addresses service level faults via the ASH technique, whereby some service level failures are handled in a more graceful fashion. Fig. 2. illustrates an ASH state-transition diagram in order to achieve service level fault tolerance. In a perfect condition, all the services are functioning correctly. The ASH MON daemon monitors service availability and health at every tunable interval and triggers alerts upon failure detection. Our current monitoring implementation is a polling approach in which a default interval is 5 seconds. However, this polling interval is tunable for the faster detection time. Section 6 entails impacts and analysis of different polling interval times.

### 3.3 Server Failover

Fig. 3 shows an example of the HA-OSCAR initial network interface configuration during a normal operation. In our example, HA-OSCAR assigns the primary server public IP address "$\mathtt{Eth0}_p$:" with `138.47.51.126` and its private IP address "$\mathtt{Eth1}_p$:" as `10.0.0.200`. We then configure an alias private IP address $\mathtt{Eth1:0}_p$: as `10.0.0.222`. For the standby server, the public IP address "$\mathtt{Eth0}_s$:" is initially unassigned and its private IP address is $\mathtt{Eth1}_s$: `10.0.0.150`.

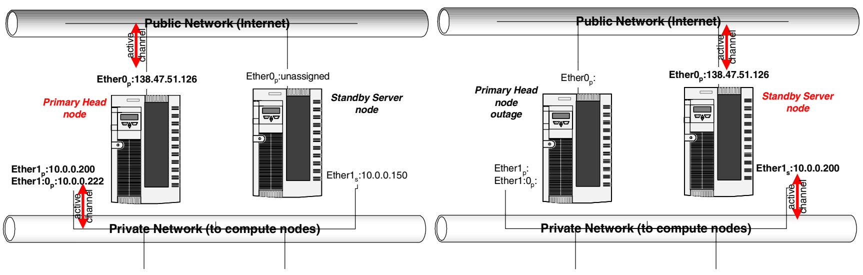


**Fig. 3.** Network Interface Configuration during normal condition and after failover

When a primary server failure occurs, all its network interfaces will drop. The standby server takes over and clones the primary server network configuration shown in Figure 3 (on the right). The standby server will automatically configure its public network interface $\mathtt{Eth0}_s$: to the original public IP address `138.47.51.126` and private network interface $\mathtt{Eth1}_s$: to `10.0.0.200`. This IP cloning only takes 3-5 seconds.

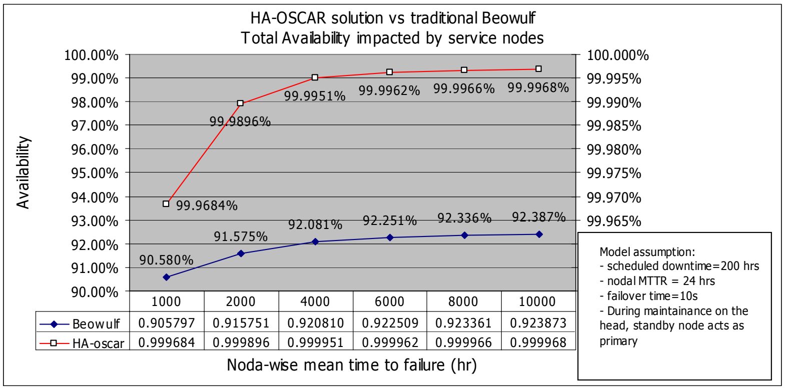
## 4 Implementation and Experimental Results

HA-OSCAR should support any Linux Beowulf cluster. We have successfully verified a HA-OSCAR cluster system test with the OSCAR release 3.0 and RedHat 9.0. The experimental cluster consisted of two dual xeon server head nodes, each with 1 GB RAM, 40 GB HD with at least 2GB of free disk space and two network interface cards. There were 16 client nodes that were also Intel dual xeon servers with 512 MB RAM and a 40 GB hard drive. Each client node was equipped with at least one network interface card; Head and client nodes are connected to dual switches as shown in Figure 1.

## 5 Modeling and Availability Prediction

In order to gauge an availability improvement based on the experimental cluster, we evaluated our architecture, its system failure and recovery with a modeling approach. We also studied the overall cluster uptime and the impact of different polling interval sizes in our fault monitoring mechanism. Stochastic Reward Nets (SRN) technique has been successfully used in the availability evaluation and prediction for complicated systems, especially when the time-dependent behavior is of great interest [5]. We utilized Stochastic Petri Net Package (SPNP) [6] to build and solve our SRN model.

We calculated instantaneous availabilities of the system and its parameters. Details can be founded in [1]. We obtained a steady-state system availability of 99.993%, which was a significant improvement when compared to 99.65%, from a similar Beowulf cluster with a single head node. Furthermore, higher serviceability such as the abilities to incrementally upgrade and hot-swap cluster OS, services, applications and hardware, will further improve planned downtime which undoubtedly benefits the overall aggregate performance. Figure 4 illustrates the total availability (planned and unplanned downtime) improvement analysis of our HA-OSCAR dual-heads vs. a single service node Beowulf clusters when exploiting redundant service nodes for both fault tolerance and hot and incremental upgrade.



| | 1000 | 2000 | 4000 | 6000 | 8000 | 10000 |
|---|---|---|---|---|---|---|
| ◆ Beowulf | 0.905797 | 0.915751 | 0.920810 | 0.922509 | 0.923361 | 0.923873 |
| □ HA-oscar | 0.999684 | 0.999896 | 0.999951 | 0.999962 | 0.999966 | 0.999968 |

Model assumption:
- scheduled downtime=200 hrs
- nodal MTTR = 24 hrs
- failover time=10s
- During maintainance on the head, standby node acts as primary

**Fig. 4.** HA-OSCAR and traditional Linux HPC: the total availability improvement analysis

## 6 Conclusions and Future Work

Our proof-of-concept implementation, experimental and analysis results [1, 2, and here] suggest that HA-OSCAR solution is a significant enhancement and promising solution to providing a high-availability Beowulf cluster class architecture. The availability of our experimental system improves substantially from 99.65% to 99.9935%. The polling interval for ASH failure detection indicates a linear behavior to the total cluster availability. The introduction of the hot-standby server is clearly cost-effective method when compared with an investment of a typical cluster since a double outage of the servers are very unlikely, therefore, clusters with HA-OSCAR are likely much more available than a typical Beowulf cluster. We have furthered our investigation from outage detection to prediction techniques [10], active-active multi-head failover and grid-enabled HA-OSCAR In addition, we have recently investigated a job queue migration mechanism based on Sun Grid Engine (SGE) and experimented with Berkley Lab Checkpoint/Restart (BLCR) and LAM/MPI [7] for an automated checkpoint and restart mechanism to enhance fault tolerance in HA-OSCAR framework. Our initial findings [9] are very promising. We will extend transparent failure recovery mechanisms for more options, including a sophisticated rule-based recovery and integration with openMPI (a unified MPI platform development effort from LA/MPI , FT/MPI [8] and LAM/MPI [7] teams).

## References

1. C. Leangsuksun, L. Shen, T. Liu, H. Song, S. Scott, Availability Prediction and Modeling of High Availability OSCAR Cluster, IEEE International Conference on Cluster Computing (Cluster 2003), Hong Kong, December 2-4, 2003.
2. C. Leangsuksun, L. Shen, T. Liu, H. Song, S. Scott, Dependability Prediction of High Availability OSCAR Cluster Server, The 2003 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'03), Las Vegas, Nevada, USA, June 23-26, 2003.
3. B Finley, D Frazier, A Gonyou, A Jort, etc., SystemImager v3.0.x Manual, February 19, 2003.
4. M J. Brim, T G. Mattson, S L. Scott, OSCAR: Open Source Cluster Application Resources, Ottawa Linux Symposium 2001, Ottawa, Canada, 2001
5. J Muppala, G Ciardo, K. S. Trivedi, Stochastic Reward Nets for Reliability Prediction, Communications in Reliability, Maintainability and Serviceability: An International Journal published by SAE International, Vol. 1, No. 2, pp. 9-20, July 1994.
6. G Ciardo, J. Muppala, K. Trivedi, SPNP: Stochastic Petri net package. Proc. Int. Workshop on Petri Nets and Performance Models, pp 142-150, Los Alamitos, CA, Dec. 1989. IEEE Computer Society Press.
7. S. Sankaran, J. M. Squyres, B. Barrett, A. Lumsdaine, Jason Duell, Paul Hargrove, and Eric Roman. The LAM/MPI Checkpoint/Restart Framework: System-Initiated Checkpointing. In LACSI Symposium, Santa Fe, NM, October 27-29 2003.
8. G.E. Fagg, E.Gabriel, Z. Chen, T.Angskun, G. Bosilca, A.Bukovsky and J.J.Dongarra: ’Fault Tolerant Communication Library and Applications for High Performance Computing’, LACSI Symposium 2003, Santa Fe, NM, October 27-29, 2003.
9. C.V Kottapalli, Intelligence based Checkpoint Placement for Parallel MPI programs on Linux Clusters, Master Thesis Report, Computer Science Program, Louisiana Tech University, August 2004 (In preparation)
10. C. Leangsuksun et al, “A Failure Predictive and Policy-Based High Availability Strategy for Linux High Performance Computing Cluster”, The 5th LCI International Conference on Linux Clusters: The HPC Revolution 2004, Austin, TX, May 18-20, 2004.