International Transactions on Systems Science and Applications
Volume 8 ▪ December 2012 ▪ pp. 13-25
sai: itssa.0008.2012.016

ISSN 1751-1461 (print)
ISSN 2051-5642 (online)

Models for Dependability Analysis of Cloud Computing Architectures for Eucalyptus Platform

J. Dantas, **R. Matos**, **J. Araujo** and **P. Maciel**

Informatics Center, Federal University of Pernambuco, Recife, Brazil

Abstract: High availability in cloud computing services is essential for maintaining customer confidence and avoiding revenue losses due to SLA violation penalties. Since the software and hardware components of cloud infrastructures may have limited reliability, fault tolerance mechanisms are means of achieving the necessary dependability requirements. This paper investigates the benefits of a warm-standy replication mechanism in three architectures based on Eucalyptus cloud computing as well as the required acquisition cost. A heterogeneous hierarchical modeling approach is used to represent the redundancy strategy and to compare the availability for the architectures presented. Both hardware and software failures are considered in the proposed analytical models. The results highlight the improvements on the availability for the redundant architectures, the respective impact on the annual downtime and the related respective costs.

Keywords: cloud computing and availability and analytical models and cost evaluation.

1. Introduction

The importance of cloud computing has been increasing for a large proportion of worldwide IT companies. Software platforms such as Eucalyptus [1] provide means to construct private and hybrid clouds in the Infrastructure as a Service (IaaS) style [2], on top of common on-premise hardware equipment [1]. Cloud services need to maintain users' confidence and avoid revenue losses but, as with any system, availability in the cloud may be affected by events such as hardware failures, planned maintenance, exchange of equipment, software bugs or updates. Fault tolerance mechanisms, such as replication techniques, are prominent approaches for coping with software and hardware limited reliability. Since cloud-based applications aim to be accessible anywhere and anytime, dependability becomes even more important and yet more difficult to achieve in such environments [3].

Hierarchical hybrid models are used in [4] to evaluate cloud-based data centers, combining reliability block diagrams and generalized stochastic Petri nets to analyze availability and reliability measures, for distinct workloads and server consolidation ratios. In [5], the authors propose a hierarchical model to analyze the availability of a Eucalyptus-based architecture tailored for e-government purposes. In [6] we used a hierarchical heterogeneous model – based on Reliability Block Diagrams (RBD) and a Markov Reward Model – to describe non-redundant and redundant Eucalyptus architectures composed of a single cluster.

This paper investigates the benefits of a warm-standby replication mechanism [7][8] in a Eucalyptus cloud computing environment, considering various combinations of numbers of clusters and redundant subsystems. A heterogeneous hierarchical modeling approach is used to represent redun-

dant architectures and compare their availability taking into account the cost of acquisition of computers, which is not considered in the other papers in this research topic. Both hardware and software failures are considered in our analytical models. These models are also used to obtain closed-form equations for computing the availability of the cloud infrastructures.

The remaining of the paper is organized as follows. Section 2 explains the background regarding analytical models and redundancy mechanisms for high availability clusters and also describes some related works. Section 3 describes the main concepts regarding Eucalyptus cloud infrastructures. Section 4 describes the case study, including the analytical models and the corresponding results. Section 5 draws some conclusions and points to possible future works.

2. Background and Related Works

System dependability can be understood as the ability to deliver a specified functionality that can be justifiably trusted [9]. An alternate definition of dependability is "the ability of a system to avoid failures that are more frequent or more severe, and outage durations that are longer than is acceptable to the user" [9]. Dependability encompasses measures such as reliability, availability, and safety. Due to the ubiquitous provision of services on the Internet and on cloud systems, dependability has become an attribute of prime concern in hardware/software development, deployment, and operation [10], since such services require high availability, stability, fault tolerance and dynamical extensibility.

There are various model types which may be used for analytical evaluation of dependability. Reliability block diagrams (RBD) [11][12], fault trees (FT) [11], stochastic Petri nets (SPN) [13] and Markov chains [14] have been used to model fault-tolerant systems and to evaluate various dependability measures. These model types differ from one to another not only in the ease of use for a particular application but in terms of modeling power [15]. They may be broadly classified into combinatorial (e.g., RBD, FT) and state-based models (e.g., SPN, Markov chains)[10]. State-based models may also be referred to as non-combinatorial, and combinatorial can be identified as non-state based models.

Combinatorial and state-space models may be hierarchically combined, in order to get the best of both worlds, e.g., a reliability block diagram is used to represent the dependability relationship between independent subsystems, while detailed or more complex fail and repair mechanisms are modeled with a Markov Reward Models [16][17]. This approach enables the representation of many kinds of dependency between components, and avoids the known issue of state-space explosion when dealing with large systems.

Systems with stringent dependability requirements demand methods for detecting, correcting, avoiding and tolerating faults and failures. A failure in a large scale system can mean catastrophic losses. Many techniques have been proposed and adopted to build failover clusters [18] as well as to leverage virtualization and cloud systems for addressing service dependability issues [19] [20]. Many of those techniques are based on redundancy, i.e., the replication of components so that they work for a common purpose, ensuring data security and availability even in the event of some component failure. Three replication techniques deserve special attention due to its extensive use in clustered server infrastructures [21] [10]:

- **Cold Standby:** The backup nodes, or modules, are turned off on standby and will only be activated if the primary node fails. The positive point for this technique is that the secondary node has low consumption of energy and do not wear the system. On the other hand, the secondary node needs significant time to be activated and clients who were accessing information on the primary node loses all information with the failure of the primary node, having to redo much of the work when the secondary node activates.
- **Hot Standby:** This type can be considered the most transparent of the replication modes. The replicated modules are synchronized with the operating module, thereby, the active and standby cluster participants are seen by the end user as a single resource. After a node fails,

the secondary node is activated automatically and the users accessing the primary node will now access the secondary node without notice the change of equipment.

- **Warm Standby:** This technique tries to balance the costs and the recovery time delay of Cold and Hot Standby techniques. The secondary node is on standby, but not completely turned off, so it can be activated faster than in the Cold Standby technique, as soon as a monitor detects the failure of the primary node. The replicated node is partially synchronized with the operating node, so users who were accessing information on the operating node may lose some information that was being written close to the moment when the primary node failed.

Most studies aimed at assessing the dependability of cloud systems do not address software dependability nor consider the influence of adding new equipments to provide redundancy for existing architectures. In [22] [23], the authors present a study on dependability prediction for servers based on the high availability cluster system known as OSCAR - Open Source Cluster Application Resources. The authors predict system reliability and availability through Stochastic Reward Nets (SRN). [24] uses continuous time Markov chains (CTMC) to analyze the availability of a cluster system with multiple nodes. Their analysis consider both common mode failure (CMF) and no CMF for the cluster nodes. CTMCs are also used in [25] for modeling HA clusters. [26] proposes a set of models for the integrated quantification of sustainability impact, cost and dependability of data center cooling infrastructures and presents a case study which analyzes the environmental impact, dependability metrics as well as the acquisition and operational costs of five real-world data center cooling architectures. [4] uses hierarchical method and proposes hybrid models combining reliability block diagrams and general stochastic Petri nets to analyze the relation between reliability and consolidation ratio, as well as the relation between availability and the workload for the virtual data center of cloud computing. In [5], the authors propose a private cloud environment suitable for e-government purposes and provide a hierarchical model to describe the proposed Eucalyptus-based architecture and compute its availability results. The hierarchical model proposed in [5] uses distinct Markov chains for cluster level and node level, while the operation level is described in a non-formal manner. In [6] we used a hierarchical heterogeneous model – based on Reliability Block Diagrams (RBD) and a Markov Reward Model – to describe the non-redundant and redundant Eucalyptus architectures. We also provided closed-form equations to compute the availability of those systems. This paper extends the previous approach by investigating the benefits of component redundancy in larger private cloud architectures, composed by two and three clusters. Other original result is the definition of closed-form expressions for computing the steady state availability of the architectures that are not presented in [6]. The closed-form expressions are obtained from the hierarchical analytical models, and enable easy and fast analysis of various scenarios. An additional contribution is the analysis of costs due to the acquisiton of equipments, based on the approach shown in [27] for data centers.

3. Eucalyptus Framework

Eucalyptus [1] implements scalable IaaS-style private and hybrid clouds [28] and is interface-compatible with the commercial service Amazon EC2 [1], [29]. This API compatibility enables one to run an application on Amazon and on Eucalyptus without modification. In general, the Eucalyptus cloud-computing platform uses the virtualization capabilities (hypervisor) of the underlying computer system to enable flexible allocation of resources decoupled from specific hardware. Eucalyptus architecture is composed by five high-level components, each one with its own web service interface: *Cloud Controller, Node Controller, Cluster Controller, Storage Controller*, and *Walrus* [28].

Figure 1 shows an example of Eucalyptus-based cloud computing environment, considering two clusters (A and B). Each cluster has one Cluster Controller, one Storage Controller, and various Node Controllers. The components in each cluster communicate to the Cloud Controller and Walrus to serve the user requests.

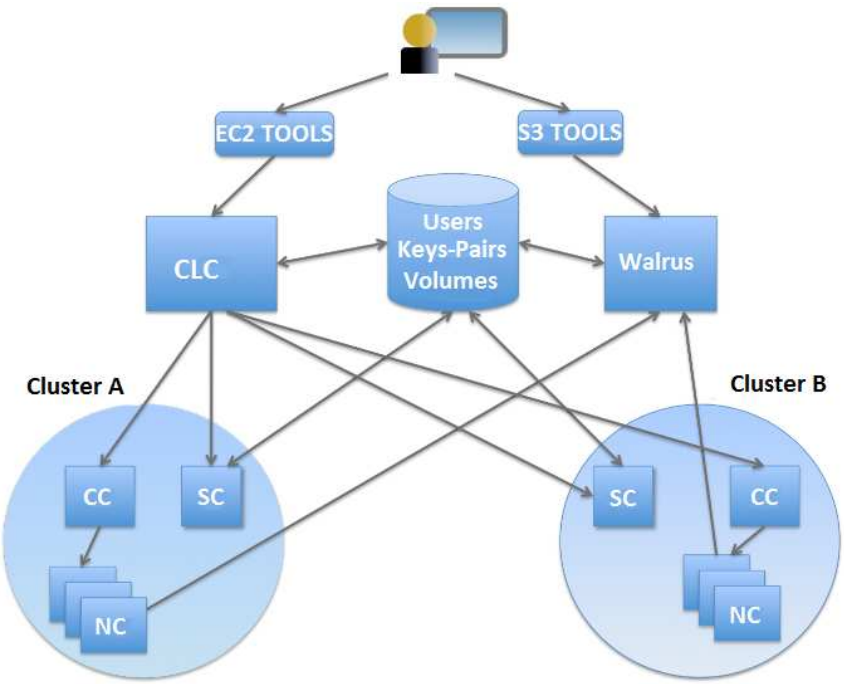


Figure 1. Example of Eucalyptus-based environment [30]

The *Cloud Controller (CLC)* is the front-end to the entire cloud infrastructure, and it is responsible for identifying and managing the underlying virtualized resources (servers, network, and storage) via Amazon EC2 API [29]. The *Node Controller (NC)* runs on each compute node and controls the life cycle of VM instances running on the node, and it makes queries to discover the node's physical resources (e.g., number of CPU cores, size of main memory, available disk space) as well as to probe the state of VM instances on that node [28], [30]. The *Cluster Controller (CC)* gathers information on a set of VMs and schedules the VMs execution on specific NCs. The CC has three primary functions: scheduling incoming requests for execution of VM instances; controlling the virtual network overlay composed by a set of VMs, gathering information about the set of node controllers, and reporting their status to the CLC [28]. The *Storage Controller (SC)* provides persistent storage to be used by virtual machine instances. It implements block-access network storage, that is similar to that provided by Amazon's Elastic Block Storage EBS [31]. The *Walrus* is a file-based data storage service compatible with Amazon Simple Storage Service (S3) [28]. The *Walrus* provides a storage service for VM images. Root filesystem as well as the Linux kernel and ramdisk images used to instantiate VMs on NCs can be uploaded to *Walrus* and accessed from the nodes.

4. Case Study

This case study analyses several possible architectures for building a redundant Eucalyptus cloud system, comparing the availability and costs of each option. The computers that composed each analyzed architecture have identical characteristics: same manufacturer, model, configuration and costs [32], shown in Table 1. Figure 2 shows a generic simple architecture which has non-redundant Eucalyptus software components and is composed of three clusters. A front end computer is adopted as the "Cloud subsystem" and configured with the Eucalyptus components known as Cloud Controller (CLC) and Walrus. Each cluster has one machine that is called hereinafter the "Cluster subsystem", which runs the Cluster Controller (CC) and Storage Controller (SC) components. Each cluster also

has three machines that run the Eucalyptus component known as Node Controller, responsible for intantiating and managing the virtual machines, interacting directly with the hypervisor. The set of three nodes in each cluster is called hereinafter a "Nodes Subsystem". Besides the architecture with three clusters, systems with one and two clusters are also evaluated in this case study. The impact of implementing the redundancy in the Cloud subsystem (CLC and Walrus) and in the Cluster subsystem (CC and SC) is considered for those systems. All the replicated architectures follow an warm-standby strategy, which may be implemented through software such as DRBD [33,34] and Heartbeat [35].

Table 1. Computer Description [32]

Brand/Model	Components	Description
DELL/Power Edge	HD	1 TB
	Memory	8 GB
	CPU	Intel Xeon - 2.2GHZ
Total cost (USD)		1339.00

Due to their simplicity and efficiency of computation, RBDs are used to analyze the dependability of the architectures. However, due to the active redundant mechanism, it was adopted a hierarchical heterogeneous model, composed of an RBD and a Markov Reward Model (MRM) representing the redundant architectures. The RBD describes the high-level components, whereas the MRM represents the components involved in the redundancy mechanism. The MRM enables to obtain a closed-form equation for the availability of the redundant subsystems.

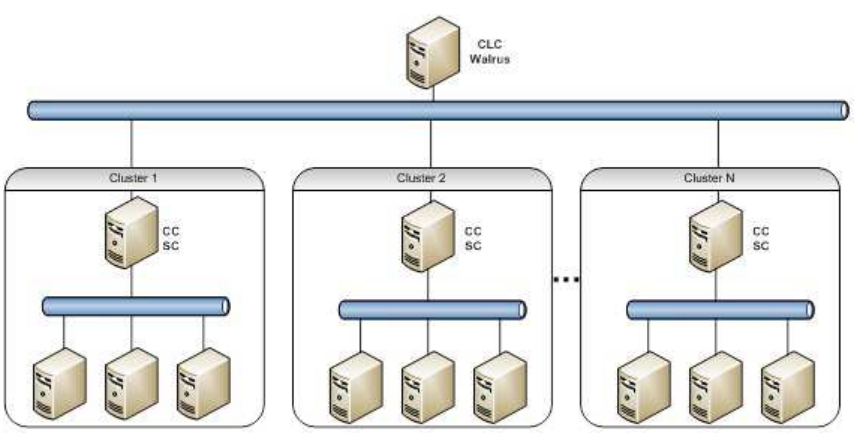


Figure 2. Private cloud architecture

4.1. Models for Non-Redundant Architectures

From a dependability point of view, the private cloud infrastructure depicted in Figure 2 may be divided in three parts: The Cloud Subsystem, the Cluster Subsystem and Nodes Subsystem. In the non-redundant architecture, the Cloud Subsystem is represented by a pure series RBD as well as the Cluster Subsystem. The Cloud Subsystem consists of hardware, operating system, and the following Eucalyptus software components: CLC (Cloud Controller) and Walrus, as shown in Figure 3.

Figure 4 depicts an RBD model for the Cluster Subsystem, which is composed of hardware, operating system, Cluster Controller (CC) and Storage Controller (SC). Table 2 presents the values of mean time to failure (MTTF) and mean time to repair (MTTR) used for the Cloud Subsystem and Cluster Subsystem models. Those values were obtained from [36] [37], and were used to compute

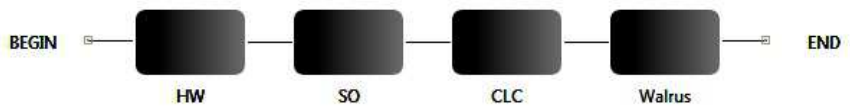


Figure 3. RBD model of the non-redundant Cloud Subsystem

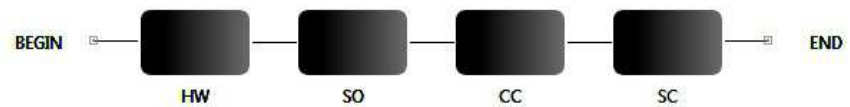


Figure 4. RBD model of the non-redundant the Cluster subsystem

Table 2. Input Parameters for the Cloud Subsystem and Cluster Subsystem

Component	MTTF	MTTR
HW	8760 h	100 min
SO	2893 h	15 min
CLC and Walrus	788.4 h	1 h
CC and SC	788.4 h	1 h

the dependability metrics for the subsystems and then for the whole system. Both Cloud and Cluster subsystems have an MTTF of 333.71 h and an MTTR of 0.93 h.

Figure 5 shows the RBD model that represents one node in the Nodes Subsystem. Besides the hardware and operating system, which are also present in the Cloud and Cluster subsystems, each node needs a hypervisor (e.g., KVM) and a Eucalyptus Node Controller in order to be available in the cloud.

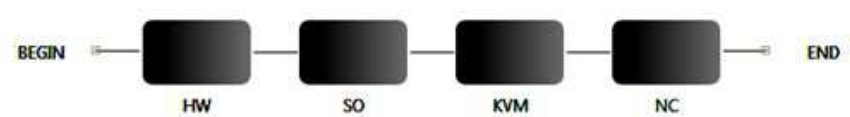


Figure 5. RBD model of one node in the Nodes subsystem

The Nodes Subsystem model assumes that the hardware and operating system of the nodes have the same dependability characteristics as in the Cloud and Cluster subsystems, i.e., the same MTTF and MTTR. Therefore, Table 3 presents only the parameter values for the KVM and NC blocks [36] [37]. The analysis of this model provides an MTTF of 481.82 hours and an MTTR of 0.91 hours for each node in this cloud environment.

Table 3. Input Parameters for the nodes

Component	MTTF	MTTR
KVM	2990 h	1 h
NC	788.4 h	1 h

RBD models were created for evaluating three scenarios which have distinct numbers of clusters and nodes and do not employ replication strategies for their main components. The Scenario I consists of one Cloud Subsystem and one Cluster Subsystem and three hosts in the Nodes Subsystem. The Scenario II is composed of one Cloud Subsystem, two Cluster Subsystems, and six hosts arranged equally in two Nodes Subsystems. The Scenario III has one Cloud Subsystem, three Clusters

Subsystems, and nine hosts divided into three Nodes Subsystems. For all scenarios, the system is available if the Cloud subsystem is running and at least one Cluster Subsystem is available, with one or more nodes running in that cluster. Figures 6, 7, and 8 show the RBD models for Scenarios I, II, and III, respectively.

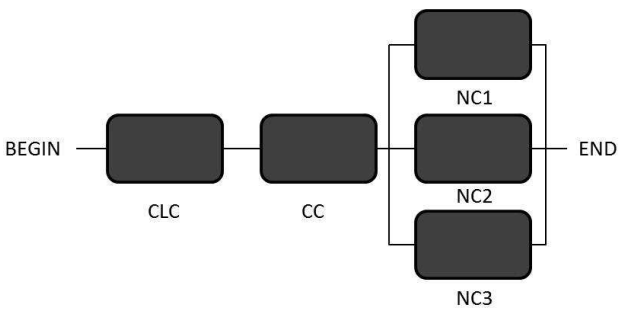


Figure 6. RBD model for the Scenario I: Cloud system with one cluster

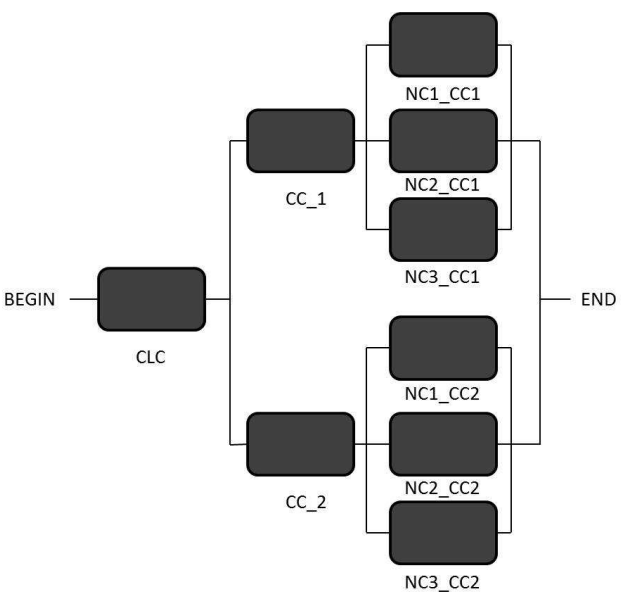


Figure 7. RBD model for the Scenario II: Cloud system with two clusters

4.2. Redundant Private Cloud Architectures

Based on the study of characteristics and components mentioned in the previous section, we proposed some failover mechanisms for achieving high availability in Eucalyptus cloud systems. We implemented a warm-standby redundant host in the main points of the system: the Cloud Subsystem and Cluster Subsystems. Software such as DRBD and Heartbeat may be used for this purpose, since they enforce the data consistency and service availability. The warm-standby replication strategy cannot be properly represented in RBD models due to dependency between states of components. Therefore, the redundant Cloud Subsystem and Cluster Subsystems are represented by Markov Reward Models (MRMs).

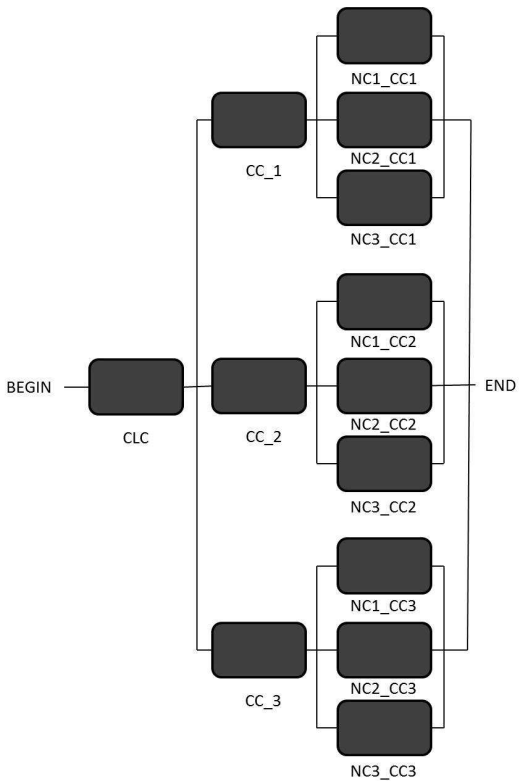


Figure 8. RBD model for the Scenario III: Cloud system with three clusters

An MRM is a labeled CTMC augmented with state reward and impulse reward structures. The state reward structure is a function ρ that assigns to each state $s \in S$ a reward $\rho(s)$ such that if t time-units are spent in state s, a reward of $\rho(s) \cdot t$ is acquired. The rewards that are defined in the state reward structure can be interpreted in various ways. They can be regarded as the gain or benefit acquired by staying in some state and they can also be regarded as the cost incurred by staying in some state. The impulse reward structure, on the other hand, is a function $\imath$ that assigns to each transition from s to s' , where $s, s' \in S$, a reward $\imath(s, s')$ such that if the transition from s to s' occurs, a reward of $\imath(s, s')$ is acquired. Similar to the state reward structure, the impulse reward structure can be interpreted in various ways. An impulse reward can be considered as the cost of taking a transition or the gain that is acquired by taking the transition.

Figure 9 depicts the MRM model which may describe both, the Cloud Subsystem and Cluster Subsystem employing warm-standby replication. In all cases, the redundant subsystem has a primary host (H1) which is active by default, and a secondary host (H2), which is the spare one. The MRM model has 5 states: UW, UF, FF, FU, and FW. In the state UW, the primary host (H1) is up and the secondary host (H2) is in a waiting condition. When H1 fails, the system goes to state FW, where H2 has not yet detected the failure of the H1. FU represents the state where H2 leaves the waiting condition and assumes the active role, whereas H1 is failed. If H2 fails before the repair of H1, the system goes to the state FF. In order to prioritize the repair of the main server there is only a single repair transition from FF, which goes to UF. If H2 fails when H1 is up, the system goes to state UF, returning to state UW with the repair of H2, or going to state FF in case of H1 also fails. The failure rates of H1 and H2 are denoted by λ_s1 and λ_s2 respectively. The rate λ_i_s2 denotes the failure rate of the secondary host when it is inactive. The repair rate of H2 is μ_s2. The transition rate sa_s2 represents the switchover rate, i.e. the inverse of the mean time to activate the secondary server after a failure of H1. Table 4 presents the values for all the mentioned parameters of the MRM. The value

of μ_s1 is equal to the value of μ_s2, the rates λ_s1 and λ_s2 also have equal values. These λ and μ values were obtained from the MTTF and MTTR computed using the RBD models for the non-redundant Cloud and Cluster subsystems. The failure rate of the secondary host when it is inactive is assumed to be 20% smaller than the failure rate of an active host. The value of sa_s2 comes from default monitoring interval and activation times found in softwares such as Hearbeat.

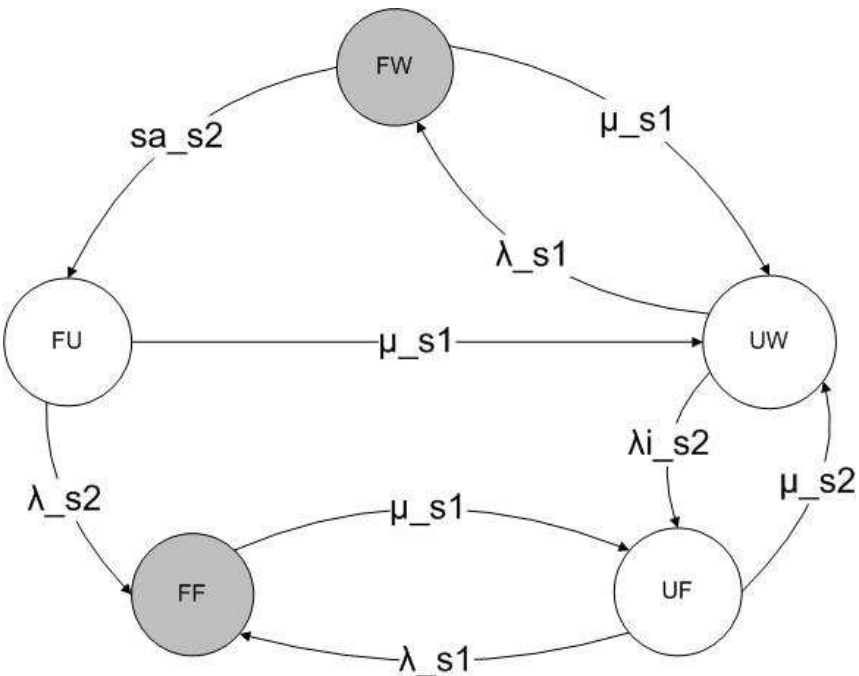


Figure 9. Markov chain model for a warm-standby redundant system with two hosts

The state reward $\rho(s)$ assigned to UW, UF and FU is equal to 1, since the (Cloud/Cluster) subsystem is available in those states. The state reward assigned to FF and FW (shaded states) is equal to 0, since the subsystem is down in those states. There are no impulse rewards in this model. Therefore, the steady-state availability of the subsystem can be computed as the steady-state reward of the MRM. Let A_{CLC} be the steady-state availability of the Cloud Subsystem, so $A_{CLC} = \sum_{s \in S} \pi_s \cdot \rho(s)$, where π_s is the steady-state probability of being in the state s, and $\rho(s)$ is the reward assigned to the state s. A similar equation may be defined for the steady-state availability of each Cluster Subsystem (A_{CC}).

Table 4. Parameter values for the Markov chain model

Parameter	Description	Value
$\lambda\_s1 = \lambda\_s2 = 1/\lambda$	Mean time for host failure	$1/333.7114$
$\lambda_i\_s2 = 1/\lambda_i$	Mean time for inactive host failure	$1/400.4537$
$\mu\_s1 = \mu\_s2 = 1/\mu$	Mean time for host repair	$1/0.938883$
$sa\_s2 = 1/sa$	Mean time to system activate	$1/0.004166$

The model of Figure 9 enables to obtain a closed-form equation for the availability of each redundant subsystem, as seen in Equation 1.

$$A_R = \frac{\mu(\lambda_i(\mu + sa) + \mu^2 + sa(\lambda + \mu))}{\lambda_i(\lambda + \mu)(\mu + sa) + \mu^2(\lambda + \mu) + sa(\lambda^2 + \lambda\mu + \mu^2)} \quad (1)$$

It is also possible to obtain a closed-form equation for computing the availability of the whole cloud infrastrucuture (A_{cloud}), from the corresponding RBD models. Equation 2 denotes how to compute A_{cloud}, according to the rule of composition of series and parallel components [38]. This is a general equation for an architecture composed of k clusters, with n nodes in each cluster. A_{CLC} is the availability of the block representing the Cloud Subsystem. A_{CC} is the availability of each Cluster Subsystem, and A_{node_i} is the availability of each node.

$$A_{cloud} = (A_{CLC}) * (1 - \prod_{j=1}^{k}((A_{CC\_j}) * (1 - \prod_{i=1}^{n}(A_{node\_i})))) \quad (2)$$

For each scenario (one, two and three clusters) mentioned in Section 4.1, three redundant variants were evaluated: (1) redundancy only in the Cloud Subsystem, (2) redundancy only in the Cluster Subsystem, and (3) redundancy in Cloud and Cluster Subsystems. The nine resulting architectures are enumerated in Table 5. Architectures A1, A2 and A3 consider the redundancy only in the Cloud Subsystem, with one, two, and three clusters respectively. Architectures A4, A5, and A6 consider the redundancy only in the Cluster Subsystem(s), for systems with one, two, and three nodes, respectively. Architectures A7, A8, and A9 consider the employment of redundancy in Cloud and Cluster Subsystems. For all architectures, the corresponding RBD model was composed with the MRM presented in Figure 9.

Table 5. Description of architectures

Architecture	Number of clusters	Description
A1	1	Redundancy only for the Cloud Subsystem
A2	2	
A3	3	
A4	1	Redundancy only for the Cluster Subsystem(s)
A5	2	
A6	3	
A7	1	Redundancy for Cloud and Cluster Subsystems
A8	2	
A9	3	

Table 6. Availability and cost measures of the system with and without redundancy

Arch.	Avail.(%)	N. of 9's	Downtime (h)	Cost (U$)
A1	99.716773	2.54786	24.81 h	8,034.00
A2	99.996535	4.46042	0.30 h	13,390.00
A3	99.997320	4.57200	0.23 h	18,746.00
A4	99.716773	2.54786	24.81 h	8,034.00
A5	99.719443	2.55197	24.58 h	14,729.00
A6	99.719443	2.55197	24.58 h	21,424.00
A7	99.994645	4.27131	0.47 h	9,373.00
A8	99.997323	4.57238	0.23 h	16,068.00
A9	99.997323	4.57239	0.23 h	22,763.00

Table 6 shows the results of this study, considering the steady-state availabilty, number of 9's [39], annual downtime, and total cost for each architecture. The architectures with highest availability

indices are A8 and A9. They correspond to the scenarios with two and three clusters, employing redundancy in both CLC and CC subsystems. This result indicates that the number of clusters does not have significant impact on the availability when the main Eucalyptus controller components (CLC and CC) are installed on redundant hosts. Due to such a similarity in availability results, the choice between architectures A8 and A9 is based on costs, which are lower for A8. It is also important to highlight the difference of about 24 hours in the downtime experienced by the "worst" architecture (A1) and the best ones (A8 and A9). Despite the lowest cost of acquistion for A1, the long unavailable time may incur other costs such as revenue losses and penalties due to SLA (service level agreement) violation.

Figures 10a and 10b enable the comparison of availability and cost results in a unified view. It is possible to observe that architectures A5 and A6 are bad choices, because the employment of redundancy does not increase the availability indices in a high degree, when compared to A2 and A3, for example, despite the costs of such actions. The existence of a single point of failure (non-redundant Cloud Subsystem) explains the poor results of architectures A5 and A6. On the other hand, Figure 10 also shows clearly that A7 has the better tradeoff, providing high availability with a relatively low cost. The difference in annual downtime from A7 to A8 and A9 is only about 14 minutes (0.24 hours), reinforcing how suitable is the architecture A7.

If only the architectures with highest computational capacity (A3, A6, and A9), i.e. those with three clusters, were considered, the better choice would be A3, because it has availability in a similar level as A6 and A9, and lower cost. This is an important result, since it indicates that for private cloud environments with 3 or more clusters, there is little benefit in employing a redundant host for the Cluster Subsystem. The redundancy in the host of Cloud Subsystem is enough to achieve a high degree of system availability.

5. Conclusions and Future Works

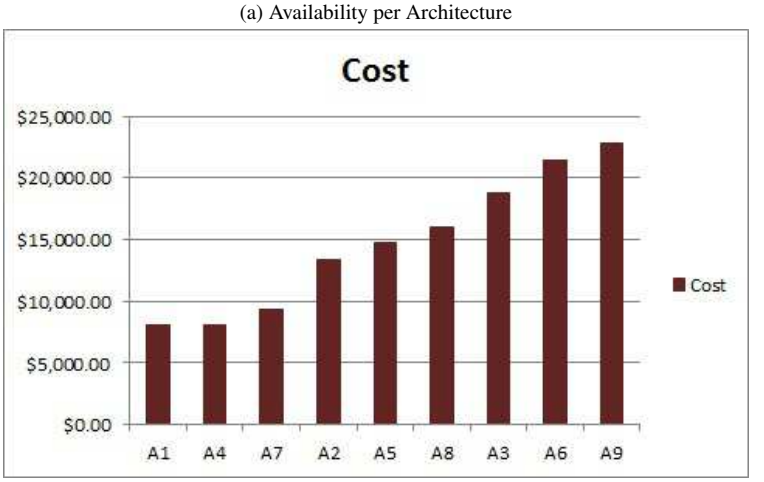
This paper investigated the benefits of a warm-standy replication mechanism in a Eucalyptus cloud computing environment. Three scenarios were used with distinct numbers of clusters and nodes. For all scenarios, the cloud system is considered available if the Cloud Subsystem is running, and at least one Cluster Subsystem is available, with one or more nodes running in that cluster. For systems with one, two, and three clusters, were proposed redundant architectures employing an warm-standby approach in the Cloud and Cluster Subsystems. The nine redundant architectures were represented by hierarchical heterogeneous models, which enabled the computation of the steady-state availability for such systems. Closed-form equations were also obtained through the analytical models, and used to compare the availability and costs of all proposed architectures, considering distinct redundancy levels. The architecture A7 has the best results, reaching 4 nines of availability (0.47 hours of downtime) using only seven devices: redundant cloud controller subsystem (two hosts), redundant cluster controller subsystem (two hosts), and three nodes. Architecures A2, A3, A8 and A9 also has 4 nines of availability, but the cost to build one of these architectures is up to twice the A7 architecture, justifying the choice for the A7 as the best one.

The approach used in this study is a guideline for adding redundancy to different private cloud architectures. It is worth mentioning that the architecture and number of equipments to be used depend on the services offered by the company. As future work, we intend to analyze, by means of experimental testbed studies, the data consistency between the replicated servers, verifying the reliability of the warm-standy redundancy mechanisms.

References

[1] Eucalyptus. Eucalyptus - the open source cloud platform. Eucalyptus Systems. http://open.eucalyptus.com/

(a) Availability per Architecture



(b) Cost per Architecture

Figure 10. Results per Architecture: Comparing Availability and Cost.

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, A view of cloud computing, Commun. ACM, Vol. 53, No. 4, Apr. 2010, pp. 50-58.

[3] D. Sun, G. Chang, Q. Guo, C. Wang, and X. Wang, A dependability model to enhance security of cloud environment using system-level virtualization techniques, Proc. First Int. Conf. on Pervasive Computing, Signal Processing and Applications (PCSPA'10), Harbin, 2010.

[4] B. Wei, C. Lin, and X. Kong, Dependability modeling and analysis for the virtual data center of cloud computing, Proceedings of the 2011 IEEE International Conference on High Performance Computing and Communications (HPCC'11), 2011, pp. 784-789.

[5] S. Chuob, M. Pokharel, and J. S. Park, Modeling and analysis of cloud computing availability based on eucalyptus platform for e-government data center, 2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS'11), July 2011, pp. 289 -296.

[6] J. Ramalho Dantas, R. Matos, J. Araujo, and P. Maciel, An availability model for eucalyptus platform: An analysis of warm-standy replication mechanism, The 2012 IEEE International Conference on Systems, Man, and Cybernetics (IEEE SMC'12), Seoul, Korea, 2012.

[7] A. Guimarães, P. Maciel, R. Matos Jr, and K. Camboim, Dependability analysis in redundant communication networks using reliability importance, Proc. of the 2011 Int. Conf. on Information and Network Technology (ICINT'11), Chennai, 2011.

[8] R. Matos Junior, A. Guimaraes, K. Camboim, P. Maciel, and K. Trivedi, Sensitivity analysis of availability of redundancy in computer networks, Proc. of The Fourth Int. Conf. on Communication Theory, Reliability, and Quality of Service (CTRQ'11), Budapest, 2011.

[9] A. Avizienis, J.-C. Laprie, B. Randell, and C. E. Landwehr, Basic concepts and taxonomy of dependable and secure computing, IEEE Trans. Dependable Sec. Comput., Vol. 1, No. 1, pp. 11-33, 2004.

[10] P. Maciel, K. S. Trivedi, R. Matias, and D. S. Kim, Dependability modeling, In: Performance and Dependability in Service Computing: Concepts, Techniques and Research Directions. Hershey: IGI Global, 2011.
[11] P. P. O'Connor and A. Kleyner, Practical Reliability Engineering, 5th ed., Wiley Publishing, 2012.
[12] B. Silva, P. Maciel, E. Tavares, C. Araujo, G. Callou, E. Sousa, N. Rosa, M. Marwah, R. Sharma, A. Shah, T. Christian, and J. Pires, Astro: A tool for dependability evaluation of data center infrastructures, 2010 IEEE International Conference on Systems Man and Cybernetics (SMC'10), Oct. 2010, pp. 783 -790.
[13] M. K. Molloy, Performance analysis using stochastic petri nets, IEEE Trans. Comput., Vol. 31, No. 9, Sept. 1982, pp. 913-917.
[14] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, Queuing Networks and Markov Chains: modeling and performance evaluation with computer science applications, 2nd ed., John Wiley and Sons, 2001.
[15] M. Malhotra, Power-hierarchy of dependability model types, IEEE Trans. on Reliability, Vol. 43, No. 2, Sept. 1994, pp. 493-502.
[16] K. S. Trivedi, M. Malhotra, and R. M. Fricks, Markov reward approach to performability and reliability analysis, Proceedings of the Second International Workshop on Modeling, Analysis, and Simulation On Computer and Telecommunication Systems (MASCOTS'94), 1994, pp. 7-11.
[17] L. Cloth, J.-P. Katoen, M. Khattri, and R. Pulungan, Model checking markov reward models with impulse rewards, Proceedings of the 2005 International Conference on Dependable Systems and Networks (DSN'05), 2005, pp. 722-731.
[18] C. Leangsuksun, L. Shen, T. Liu, H. Song, and S. Scott, Dependability prediction of high availability oscar cluster server, Proceedings of the 2003 Int. Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA'03), Las Vegas, Nevada, USA, June 2003.
[19] W.-L. Yeow, C. Westphal, and U. Kozat, A resilient architecture for automated fault tolerance in virtualized data centers, IEEE Network Operations and Management Symposium (NOMS'10), 2010, pp. 866 -869.
[20] V. Chaudhary, M. Cha, J. Walters, S. Guercio, and S. Gallo, A comparison of virtualization technologies for hpc, 22nd Int. Conf. on Advanced Information Networking and Applications (AINA'08), 2008, pp. 861-868.
[21] I.-R. Chen and F. Bastani, Warm standby in hierarchically structured process-control programs, Software Engineering, IEEE Transactions on, Vol. 20, No. 8, Aug 1994, pp. 658 -663.
[22] C. Leangsuksun, L. Shen, T. Liu, H. Song, and S. Scott, Availability prediction and modeling of high availability oscar cluster, Proceedings of the IEEE International Conference on Cluster Computing (Cluster'03), Hong Kong, December 2003.
[23] C. B. Leangsuksun, L. Shen, T. Liu, and S. L. Scott, Achieving high availability and performance computing with an ha-oscar cluster, Future Generation Computer Systems, Vol. 21, No. 4, 2005, pp. 597 - 606.
[24] Z. Hong, Y. Wang, and M. Shi, Ctmc-based availability analysis of cluster system with multiple nodes, in Advances in Future Computer and Control Systems, Vol. 160, 2012, pp. 121-125.
[25] C. Leangsuksun, L. Shen, H. Song, S. Scott, and I. Haddad, The modeling and dependability analysis of high availability oscar cluster system, The 17th Annual International Symposium on High Performance Computing Systems and Applications (HPCS'03), May 2003.
[26] G. Callou, P. Maciel, D. Tutsch, and J. Arajo, Models for dependability and sustainability analysis of data center cooling architectures, The 2nd International Workshop on Dependability of Clouds, Data Centers and Virtual Machine Technology (DCDV'12) in conjunction with The 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'12), Boston, MA, USA, 2012.
[27] J. Figueiredo, P. Maciel, G. Callou, E. Tavares, E. Sousa, and B. Silva, Estimating reliability importance and total cost of acquisition for data center power infrastructures, 2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Oct. 2011, pp. 421 -426.
[28] Eucalyptus, Eucalyptus Open-Source Cloud Computing Infrastructure - An Overview, Eucalyptus Systems, Goleta, CA, 2009.
[29] Amazon. Amazon Elastic Compute Cloud - EC2. Amazon.com, Inc. http://aws.amazon.com/ec2/
[30] J. D, K. Murari, M. Raju, S. RB, and Y. Girikumar, Eucalyptus Beginner's Guide, 2010.
[31] Amazon. (2011) Amazon Elastic Block Store (EBS). Amazon.com, Inc. http://aws.amazon.com/ebs
[32] Dell, Dell computers. http://www.dell.com/
[33] DRBD, Distributed replicated block device. http://www.drbd.org/
[34] P. Reisner, DRBD - distributed replicated block device, Proc. of the 9th Int. Linux System Technology Conference, Cologne, Sept. 2002.
[35] Heartbeat, Linux-HA. http://www.linux-ha.org
[36] D. S. Kim, F. Machida, and K. Trivedi, Availability modeling and analysis of a virtualized system, 15th IEEE Pacific Rim Int. Symp. on Dependable Computing (PRDC'09), 2009, pp. 365-371.
[37] T. Hu, M. Guo, S. Guo, H. Ozaki, L. Zheng, K. Ota, and M. Dong, MTTF of composite web services, 2010 Int. Symp. on Parallel and Distributed Processing with Applications (ISPA'10), Sept. 2010, pp. 130 -137.
[38] W. Kuo and M. Zuo, Optimal reliability modeling: principles and applications. John Wiley & Sons, 2003.
[39] M. Marwah, P. Maciel, A. Shah, R. Sharma, T. Christian, V. Almeida, C. Araújo, E. Souza, G. Callou, B. Silva, S. Galdino, and J. Pires, Quantifying the sustainability impact of data center availability, SIGMETRICS Perform. Eval. Rev., Vol. 37, Mar. 2010, pp. 64-68.