International Journal of Computer Science Trends and Technology (IJCST) – Volume 4 Issue 5, Sep - Oct 2016

RESEARCH ARTICLE OPEN ACCESS

Building a Low Cost Energy Efficient Private Cloud

P. Jayanthi [1], E. Prakash Babu [2], V. Janardhan Babu [3]
M.Tech [1]. Assoc. Professor [2], Professor [3]
Department of Computer Science and Engineering
SVPCET, Puttur
Andhra Pradesh - India

ABSTRACT

In moderate enterprise organizations who offers IaaS cloud services, addressing individual component failures is of complex activity. It further leads to violation of Service Level Agreements (SLAs) on the availability of cloud services. Service provider needs to scrutinize the analytical reports of service failures and then to design a system capable of sustaining the SLAs. The most popularly used monolithic model is simulated here to address the various performance parameters.

Keywords:— Energy efficient, Private cloud, availability; downtime, cloud computing, simulation.

I. INTRODUCTION

Infrastructure as a Service (IaaS) is a form of cloud computing used to offers virtualized computing resources to the agricultural storage needs over the Internet. An IaaS model is built with hosts, software's, servers, storage and other infrastructure components to meet the needs that arise from time to time. The proposed resilient IaaS model host applications and handle tasks including system maintenance, backup and resilience planning. Infrastructure-as-a-Service (IaaS) is emerging as an easily deployable service with flexibility and cost-effectiveness anticipate and evolve with the stake holders' rapidly changing business requirements. The proposed cloud provides simple provisioning of processing, storage, network, and other fundamental computing resources over a network by configuring computing resources that meets the demands of dynamic loads. The IaaS cloud provides virtual machines, allocates storage, dynamically configures load balancers and allocates IP addresses in an efficient manner. The proposed architecture ensures the 24x7 service to all the stack holders and hence increases information sharing in quick time with ever change of load that is temporary, experimental or change unexpectedly. Further the automation of administrative tasks, dynamic scaling, desktop virtualization and policy-based services are built over the proven core modules of eucalyptus platform. Typical IaaS cloud services offered among the stakeholders are illustrated in the figure shown below.

Figure 1 Storage and infrastructure sharing across the cloud users

In a typical large Infrastructure-as-a-Service (IaaS) cloud environment, component failures are quite common. Such failures may lead to occasional system downtime and eventually fails to meet the Service Level Agreements (SLAs) promised by the cloud service providers. The availability analysis of the underlying infrastructure is not only useful to the service provider in designing a system capable of providing a defined SLA, but also to evaluate the performance analysis of the existing infrastructure. This proposal aims to quantify the performance of an existing large-scale IaaS cloud, where failures are typically dealt through migration of physical machines among pools: running, standby (turned on, but not ready), and Spare (turned off). The Virtual machines are created mapped over Physical Machines will be Migrated Based on Guard conditions to provide uninterrupted on demand services.

The current research is focused on evaluating a scalable, stochastic model-driven approach that quantifies the availability of a large-scale IaaS cloud, where failures are typically dealt with migration of virtual machines among three pools. Improved monolithic model is integrated into the cloud infrastructure to carry the complexity analysis and affordable solution is provided with energy efficiency. Dependencies among them pools are resolved using MTBF values are drawn from the manufacturer's specifications for which a solution is provided. The analytic-numeric solutions obtained from the earlier research and improved monolithic models needs to be applied for performance measures and will be compared. The solution is also considered for the proposed model, and solution times of the methods are compared.

II. OBJECTIVES

The objectives of this paper are

- To mitigate the knowledge sharing among the stake holders a round the clock solution is proposed to access the built services with energy efficiency.
- The models and approaches developed with this proposal will be highly useful to any Cloud service provider offering IaaS services to meet their SLAs.

- Planning, forecasting, and detection of bottlenecks while carrying the what-if analysis or overall optimization of cloud infrastructure.
- To develop an energy efficient mechanism to reduce operational costs of the IaaS Cloud provider.
- The proposal presents vision, challenges, and architectural elements for energy-efficient management of Cloud computing environments

III. NATIONAL AND INTERNATIONAL STATUS

International Status:

Availability Modeling and Analysis of Virtualized System[1] was in continuous development. Virtualized two hosts system models using a two-level hierarchical approach were developed using homogeneous continuous time Markov chains (CTMC). The models incorporate not only hardware failures (e.g., CPU, memory, power, etc) but also software failures including Virtual Machine Monitor (VMM), Virtual Machine (VM), and application failures. The evaluations Metrics are steady state availability, downtime in minutes per year and capacity oriented availability.

Markov and Markov reward models [2] are widely used for the performance and reliability analysis of computer and communication systems. Models of real time systems often contain thousands or even millions of states using Stochastic Reward Nets (SRNs) for the automatic generation of these large Markov reward models.

High availability is one of the key characteristics of Infrastructure-as-a-Service (IaaS) cloud. In Scalable Availability Model [3], The authors presents a scalable method for availability analysis of large scale IaaS cloud services using analytic models to reduce the complexity of analysis and the solution time.

Handling diverse client demands and managing unexpected failures without degrading performance are two key promises of a cloud delivered service. However, evaluation of a cloud service quality is cumbersome as the scale and complexity of cloud system increases. Service availability and provisioning response delays are two key QoS metrics. A novel approach is proposed to reduce the complexity analysis by dividing the overall model into sub-models and then a solution is obtained by iteration over individual sub-model solutions.

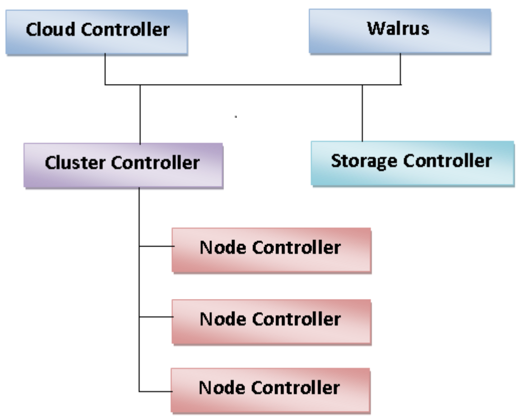

Figure 2 Eucalyptus architecture

National Status

High availability cloud model are built and compared availabilities models that differ in the sixth decimal place during the design phase, fairly detailed stochastic models are developed and evaluated the design and perform design tradeoffs. Modeling High Availability systems [2] using three-level hierarchical decomposition that mixes reliability block diagrams and Markov chains. The model is built and evaluated using the SHARPE software package.

Modern day datacenters host hundreds of thousands of servers that coordinate tasks in order to deliver highly available cloud computing services. These servers consists of multiple hard disks, memory modules, network cards , processors etc., each of which while carefully engineered are capable of failing. While the probability of seeing any such failures in the lifetime (typically 3-5 years in industry) of a server can be somewhat small, these numbers get magnified across all devices hosted in a datacenter. At such a large scale, hardware component failure is the norm rather than an exception.

To the best of our knowledge, cloud computing Hardware Reliability [3] is the first attempt to study server failures and hardware repairs for large datacenters. A detailed analysis is presented on failure predicators.

Methodology

Phase 1: Building a secured private cloud server

Phase 2: Designing and validating the Fault Tolerant Modules with high performance

- **Cloud Controller Interface Module :**
  This module helps the cloud administrator in configuring and underlying computing, storage and network storages.
- **Cluster/Node Controller's performance tuning**
- **Storage Controller**
  To offer block level persistence storage for all the virtual machines launched in the cloud.
- **Cloud Watch**
  To monitor the functioning of virtual machines, Scheduled and Running jobs in the Virtual Cloud.
- **VM Log recorder and analyzer**
- **Improved monolithic model design**(Migration of

PMs, Repair policy of PMs)

This module offers high switching of virtual machines across the pools as per the needs of number of jobs queued in the cloud server.

- **Load balancing JOB Scheduler**

  This kind of elastic load balancer that allocates all the jobs queued in cloud for running.

- **Mapping of Infrastructure to virtual machines and pools**

  Installing the cloud controller and storage control and integrating the functionalities to offer creation of virtual cloud services.

Phase 4: Implementation of Modules

Phase 5: Integrating the cloud services with fault-tolerant modules

Phase 6: Testing

Modules Description:

1. Monolithic accessibility model
2. Model Outputs
3. Repair Module

3.2.1 Monolithic accessibility model:

Fig. 1 IaaS cloud accessibility analysis shows that the model for the SRN is a monolith. Input parameters for such a model are: 1) the initial number of PMS in each pool (NH, NW, and NC), 2), hot, and the cold of PMS MTTFs (1 = h, 1 = w and 1 = c, respectively) Each pool (NR) of a PM, 4) MTTR (1 = m) repair facilities, 3) the number, and the pools of PMS 5) MTTMs (1 = GWh; 1 = gch; 1 = ghw; 1 = GCW; 1 = ghc; 1 = GWC). NH, NW, NC, and NR design parameters, MTTF, MTTR, and MTTM values are measured experimentally. Warm Table 1. Hot Model guard duties, and a cool pool Ph, PW, and PC, respectively, and with the number of tokens in the pool, such as the number of non-represented positions P have failed. Heat the event of failure transitions PM Tbwhf, Tbchf, and represents Thf fi ring. PM fails to migrate a PM from a different pool of hot and three cases arise: a warm PM heated pool (Tbwhf fi res) to the transmission can be avail- 1), 2) the warm pool was empty, but a cold PM (Tbchf fi res) can be migrated and 3) both warm and cold pool) failed to heat the space and any other PM PM (Thf fi res are unable to be replaced. The three mutually exclusive cases [G2] [G1] guard duties are by the pattern and. the amount of heat available for PM fail- ure rate So multiply that by the number of P h should be equal to the heat, the rates reported in Table 2. # transitions can be considered depending on the number of tokens in the input Ph place near the map

The rates depend on the figuring arcs are used to represent such a marking. Pwhm places tokens; Pchm, and wait until the end of colonial Pcwm heated pool, refer to p. In particular, the transition Tbwhf fi res, PW and Ph places a token for each one is taken from the hot pool as a token borrowed a warm PM modeling, Pwhm put in place. Subsequently, the transition Twhm (full migration) Fi on the ring, a token and a token is removed from the places Pwhm pH and each deposited Pbw. Twhm rate of transformation performed in parallel to all the immigrants P migration process modeling is based on the number of tokens in Pwhm. Place Pbw repaired and returned at the end of the repair process, the number of P failed to keep track of the warm pool. Similarly, the transition to fiction Tbchf on the ring, a token, a token is deposited to Pchm places each and will be removed from the PC and Ph.

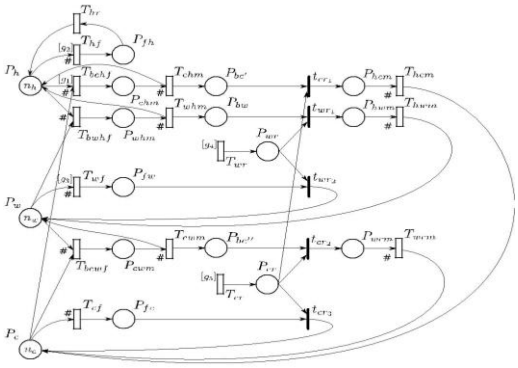

Fig. 1. Monolithic SRN model for accessibility analysis of IaaS cloud.

Guard functions	Values
[g1]	1 if ¿P_w=0 0 Otherwise
[g2]	1 if ¿P_w=0∧¿P_c=0 0 Otherwise
[g3]	1 if ¿P_c=0 0 Otherwise
[g4]	1 if ¿P_{fw}+¿P_{bw}>0 0 Otherwise
[g5]	1 if ¿P_{fc}+¿$P_{bc'}$+¿$P_{bc''}$>0 0 Otherwise

TABLE 1:Guard Functions for Monolithic and Interacting SRN Sub-Models

Transitions	Rates of transitions
T_{hf}	$\#P_h \cdot \lambda_h$
T_{bchf}	$\#P_h \cdot \lambda_h$
T_{bwhf}	$\#P_h \cdot \lambda_h$
T_{wf}	$\#P_w \cdot \lambda_w$
T_{bcwf}	$\#P_w \cdot \lambda_w$
T_{cf}	$\#P_c \cdot \lambda_c$

TABLE 2: Rates of Transitions Modeling the Failure of PMsin Monolithic SRN Model

Tchm the transition to fiction on the ring, a token and a token is removed from the Pchm depos- ited places will each pH and Pbc0. Tchm conversion rate is based on the number of tokens in Pchm. Repairs and renovations at the end of the pro cess Pbc0 needs to be given back to the cold pool failed to keep track of the number of P. Thf transition fi res, a token reduction of PMS modeling available to a place by the heated pool and a token is removed from PH repaired and given back to the pool to be heated to represent PM in place of the failed when they were deposited in Pfh. Failure in the warm pool, repair, and PMS immigrants are represented in the same way. Tbcwf transitions in the event of failure of a warm PM and will be made by TWF. A cold PM, warm pool, the transition Tbcwf fi res migration if available, to cool the pool is empty, TWF fi res. Mutual exclusion between the two cases [G3] is guaranteed by the guard func- tion. Throughout the warm PM w must be equal to the failure rate multiplied by the number of available P so warm, transitions Tbcwf and the TWF considered standard based on the number of tokens in place of the PW reported in Table 2, when the transition Tbcwf Fi res, taken from the PC as a token in place and keep it accumulated Pcwm. Until the migration is complete, place the warm pool Pcwm, are kept in cold P hold. Tcwm the transition to fiction on the ring, a token and a token is removed from the places Pcwm PW and deposited to Pbc00. PM has failed in two models, repaired and re-migrate to the cold pool. TWF-Fi on the ring, place a token moves from PW to PFW. PM has failed repaired and back in the designs migrate to the warm pool. TCF transformation models the number of tokens on the PC in a cool place for PM dependent on the failure rate of its jobs (see Table 2). In this way, the cold PM failure rate is equal to C multiplied by the number of available P cool. TCF transition fi res, place a token moves from PC when Pfc place. Failed to repair the rates for each pool of PMS NR repair facilities in order to model the presence of the marking on the THR, Twr, and TCR (Table 3 see) represents the transitions.

Transitions	Rates of transitions	
T_{hr}	$\#P_{fh}.\mu$	$if\ \#P_{fh} \le n_r$
	$n_r.\mu$	$Otherwise$
T_{wr}	$(\#P_{fw} + \#P_{bw}).\mu$	$if\ \#P_{fw} + \#P_{bw} \le n_r$
	$n_r.\mu$	$Otherwise$
T_{cr}	$(\#P_{fc} + \#P_{bc'} + \#P_{bc''}).\mu$	$if\ \#P_{fc} + \#P_{bc'} + \#P_{bc''}$
	$\le n_r$	
	$n_r.\mu$	$Otherwise$

TABLE 3:Rates of Transitions Modeling the Repair of Failed PMs in Monolithic SRN Model and Interacting SRN Sub-Models

Twr and TCR transitions are enabled only at least a PM needs to be repaired. The guard duties [G4] and [G5] is determined. As soon as the transitions, twr2, tcr1, tcr2 twr1, and the repair process to start immediately after the completion of the migration, the fact that the model of PMS is repaired pool tcr3. Places Phcm, Phwm, and repaired migration Pwcm model holding of PMS. Migra- tion, depending on the changes in the rates of fi ring Thcm, Thwm, and will be made by Twcm. Table 4, we summarize the rates of PMS migration modeling of all transitions.

3.2.2 Model outputs:

SRN level assigned to the desired function of the rate of reward and the reward rate is estimated to be valued in stable condition [1]: a Markov model of the reward system is used to measure outputs. Interest Our actions are as follows:

In each pool of PMS

(i) the number of Mean. , Hot, hot and cold in the pool but failed to replace the Ph of the average number of PMS given by the average number of tokens, PW and PC (E½ # Ph, E½ # PW and E½ # Pcin indicated) following. These actions are summarized in Table 3 for the gift of doing things.

(ii) cloud service (A) accessibility. We are more than the total number of PMS in the hot pool, or (1 k NH with) k equally considered available if the cloud service.

Measures	Reward rates
Mean number of PMs in the hot pool($E[\#P_h]$)	$\#P_h$
Mean number of PMs in the warm pool($E[\#P_h]$)	$\#P_w$
Mean number of PMs in the cold pool($E[\#P_c]$)	$\#P_c$
Accessibility of cloud service(A)	1 if $\#P_h \ge k$; 0 o/w
Probability to have at least one PM in warm pool(P_w)	1 if $\#P_w \ge k$; 0 o/w
Probability to have at least one PM in cold pool(P_c)	1 if $\#P_c \ge k$; 0 o/w

TABLE 4: Reward Rates to Compute Different Output Measures from Monolithic SRN Model

3.2.3 Repair Module

They represent a large variety of clouds can be solved for our models in a wide range of parameter values. However, some interesting results we report in this paper. MTTF 100-500 hours may be in the range of PMS is not hot, warm MTTF 500- 2,500 hours of PMS should be in the range of 300- 1,750 hours, and cool to be in the range of PMS MTTF. The repair process may vary depending on the type of a PM's MTTR:

(i) The software is based on the fully automated repair (1-30 minutes),

(ii) completely manual repair (1- 5 days) and

(iii) a combination of manual and automated repair (1-12 hours) are assumed to vary between a PM's MTTMs 10-60 minutes. 0 GHz CPU and 4 GB memory: All models have 3 can be solved by using a desktop PC. Thus, the reported figures relative to the machine, but we believe that similar trends are also using other computers.

IV. CONCLUSION

This project is a large (infrastructure-as-a-service) IaaS cloud systems with a Monolithic model approach for the analysis of accessibility. Interacting with the simulation models for the spread of the issues addressed by the Deputy to show how. Without significantly compromising the accuracy of interacting sub-models to quickly provide solutions to the spread of the utility model approach. For larger systems close to the simulation, the results are obtained faster, monolithic and interacting with the sub-models and the results of the match. In special cases, closed form solutions to solve quickly, producing a large cloud models. Cloud IaaS cloud service providers design, develop, test, and operation can benefit from the proposed modeling approach. Design and development time, providers are required to provide a specific accessibility SLA can use these models to determine the size of the pool. And operational phases of testing, repair strategies for providers to dynamically tune the parameters (eg, manual vs. automated parallel Repair Repair, No), the promise can maintain the accessibility of the SLA.

REFERENCES

[1]. Kim. D, F. Machida, and K.S. Trivedi, “Availability Modeling and Analysis of a Virtualized System,” Proc. 15th IEEE Pacific Rim Int’l Symp. Dependable Computing, 2009.

[2]. Ciardo G. et al., “Automated Generation and Analysis of Markov Reward Models Using Stochastic Reward Nets,” Mathematics and Its Applications: Linear Algebra, Markov Chains and Queueing Models, vol. 48, pp. 145-191, Springer, 1993.

[3]. Francesco Longo, Rahul Ghosh, Vijay K. Naik, and Kishor S. Trivedi, “A Scalable Availability Model for Infrastructure-as-a-Service Cloud”, 2011, Dependable Systems & Networks (DSN), 2011 IEEE/IFIP 41st International Conference.

[4]. Rahul Ghosh, Kishor S. Trivedi, Vijay K. Naik and Dong Seong Kim “End-to-End Performability Analysis for Infrastructure-as-a-Serice Cloud: An Interacting Stochastic Models Approach”, 2010, Pacific Rim International Symposium on dependable computing.

[5]. Trivedi.K.S., R. Vasireddy, D. Trindade, S. Nathan, and R. Castro, “Modeling High Availability System,” Proc. 12th Pacific Rim Int’l Symp. Dependable Computing, 2006.

[6]. Trivedi .K.S. and R. Sahner, “SHARPE at the Age of Twenty Two,” ACM SIGMETRICS Perf. Eval. Rev., vol. 36, no. 4, pp. 52-57, 2009.

[7]. K.Viswanath and N. Nagappan, “Characterizing cloud computing Hardware Reliability,” Proc. First ACM Symp, Cloud Computing(SOCC) 2010.

[8]. V. Mainkar and K.S. Trivedi, “Sufficient Conditions for Existence of a Fixed Point in Stochastic Reward Net-Based Iterative Mod- els,” IEEE Trans. Software Eng., vol. 22, no. 9, pp. 640-653, Sept. 1996.

[9]. L. Tomek and K.S. Trivedi, “Fixed-Point Iteration in Availability Modeling,” Proc. Fifth Int’l GI/ITG/GMA Conf. Fault-Tolerant Computing Systems, Tests, Diagnosis, Fault Treatment, vol. 91, pp. 229-240, 1991.