Preventive Maintenance for Virtualized Local Disaster Recovery Plan

Aye Myat Myat Paing, Ni Lar Thein
University of Computer Studies, Yangon
paing.ayemyat@gmail.com

Abstract

Preventive maintenance combined with disaster recovery planning will help to minimize the hardware and software problems if disaster occurs. Disaster recovery processes involve unplanned interruption of services. Unplanned downtime is mainly caused by software failure due to software aging rather than hardware failure. Preventive maintenance (software rejuvenation) is used to counteract the software aging phenomenon. In this paper, we propose preventive maintenance model for virtualized local disaster recovery plan through a stochastic Petri net model. In the proposed model, active-standby virtualized clustering architecture is employed. We analyze how preventive maintenance can improve the system availability of virtualized local disaster recovery plan. We perform the evaluation of the proposed model using SHARPE simulation tool.

Keywords: availability, clustering, local disaster recovery, preventive maintenance, stochastic Petri nets, virtualization.

1. Introduction

As businesses increasingly rely on IT for their mission-critical operations, it is essential to have plans in place to ensure your business viability is not at risk from a critical incident. Business continuity is the ultimate goal of critical facilities. The disaster recovery (DR) is a critical part of business continuity, refers to the process of restoring mission-critical systems, applications and data at time of interruption. There are two broad categories such as natural or man-made disaster which can cause site failures. There are many different options for disaster recovery such as utilizing recovery site services which can bring your business back up to speed quickly. Therefore disaster recovery planning is essential. A good disaster recovery plan covers the hardware and software required to run critical business applications in the event of disaster.

Disaster and its recovery processes involve unplanned interruption of service. Unplanned downtime is mainly caused by computer failure, network failure, software failure and local or regional disaster [8]. Preventive maintenance (PM) refers to the schedule of planned maintenance actions aimed at the prevention of breakdowns and failures in the event of disaster. The goal of various preventive maintenance tasks to software and hardware, disaster recovery procedures and networking monitoring is to enhance reliability and performance.

As business becomes increasingly dependent on information and computing technology, continuous availability is a universal concern. Failures of computer systems are more often due to software faults than due to hardware faults. The state of software degrades with time is known as software aging. Software aging has not only been observed in software used on a mass scale but also in specialized software used in high-availability and safety critical applications. The most effective way to handle software failure due to software aging is software rejuvenation. Software rejuvenation is a proactive fault management technique aimed at cleaning up the system internal state to prevent the occurrence of more severe crash failures in the future [13]. This process removes the accumulated errors and frees up operating system resources, thus preventing in a proactive manner the unplanned and potentially expensive system outages due to the software aging. The necessity to do preventive maintenance, not only in general purpose software systems of mass use, but also in safety-critical and highly available systems. Since the preventive action can be done at optimal times, it reduces the cost of system downtime compared to reactive recovery from failure [11].

Maximizing the availability of computer systems and services is becoming the primary focus in IT environments today [4]. Traditional disaster recovery solutions require a great deal of duplicate hardware and software. Virtualization affords significant cost and performance advantages over more traditional disaster recovery options. A virtualization layer is a software layer that abstracts the physical resources for use by the Virtual Machines (VMs). Virtual machine technologies require availability solutions that provide protection against data loss and downtime for the entire environment. The majority of today's high availability cluster is based on real physical hardware and virtualization is coming more and more popular nowadays, one has to think about possible combinations of virtualization and high availability clustering in DR environment [1].

Analytical models are mathematical models which are an abstraction from the real world system and relate only to the behavior and characteristics of interest. A Petri Net (PN) is a graphical paradigm for the formal description of the logical interactions among parts or of the flow of activities in complex systems. A complementary issue is to specify the system behavior in a concise way from which the underlying stochastic process can be extracted and analyzed. Petri nets with their remarkable flexibility and potential for capturing concurrency, contention and synchronization in a system have been widely used for qualitative modeling [9], [10].

In this paper, we combine clustering technology, virtualization technology and preventive maintenance (such as software rejuvenation) for virtual machine servers in order to improve the DR performance. We have considered the software failure due to aging problem in the virtualized local disaster recovery plan through the use of preventive maintenance action. We construct a stochastic Petri net model of preventive maintenance for virtualized local disaster recovery plan to analyze the effectiveness of the proposed method that we combine technology. We also analyze the preventive maintenance model with simulation using SHARPE (Symbolic Hierarchical Automated Reliability and Performance Evaluator) tool.

The rest of this paper is organized as follows. Section 2 presents related work. Modeling of the proposed PM model for virtualized local disaster recovery plan, and analysis of the model are described in section 3. Finally, we conclude the paper in section 4.

2. Related Work

Clitherow et al. [3] discussed high availability and disaster recovery solutions, and described how HA and DR solutions differ from one another and how they can be combined to provide the highest levels of resiliency for IT infrastructures. The primary objective of the paper [6] was how to help ensure college business continuity by providing the ability to successfully recover computer services in the event of a disaster. They summarized the results of a comprehensive risk analysis conducted for all IT services; they provided general steps that will be taken in event of a disaster to restore IT functions and also provided the preventive maintenance for hardware, software and network monitoring system.

In [2] they discussed the development, maintenance and testing of the disaster recovery process. This plan provided recovery procedures to be used at the present data center site after repairs have been made or at the Disaster recovery Backup Site. In [1] they discussed how combined with clustering, server virtualization and intelligent network storage enable IT departments to employ DR sites with active-active architectures.

Salfner et al. [5] presented and analyzed a coloured stochastic petri net model of a redundant fault-tolerant system. They have simulated the Petri net model for different levels of utilization and have computed service unavailability in the different model configurations.

Garg et al. [11] presented a model for a transactions based software system which employs preventive maintenance to maximize availability, minimize probability of loss, minimize response time, or optimize a combined measure. They evaluated the three measures for two different preventive maintenance policies and showed via numerical examples. In [12] presented and analyzed three software rejuvenation policies for an operational software system with multiple degradations using stochastic Petri nets and continuous–time markov chains are used to describe the analytic models.

3. Proposed PM model for Virtualized Local DR Plan

Disaster Recovery Plan resumes the IT systems. The objective of a disaster recovery plan is to restore the operability of systems that support mission-critical and critical business processes to normal operation as quickly as possible. Disaster recovery recovers operations when the primary datacenter site fails completely and / or the high availability mechanisms can no longer maintain application availability. With disaster recovery in place, organizations can resume operations at a secondary site. Local disaster recovery plan is that the surviving node can support the service for a failed node in the event of localized disaster such as floods, fire or building power outages.

In this section, we describe the proposed virtualized local disaster recovery (VLDR) plan is based on active-standby virtualized clustering architecture as shown in Figure 1. Clustering supports two or more servers running duplicate VMs. Failover technologies that allow a VM on a troubled server to migrate seamlessly to an available server are also available. Failover technologies also allow a failed VM to load from a storage snapshot and start up on another server. Each service under availability DR solution needs at least two sites: a primary site, on which the service run, and disaster recovery site, able to recover the application. The active physical server at primary site as well as the standby physical server at DR site contains two or more VMs. At primary site, VMs are created as active VMs. At the same time, VMs are created as standby VMs at DR site. As a result of failure detection, the active-standby roles are switched. We employ shared storage for the active-standby clustering architecture. A heartbeat keep-alive system is used to monitor the health of the nodes between primary and DR site.

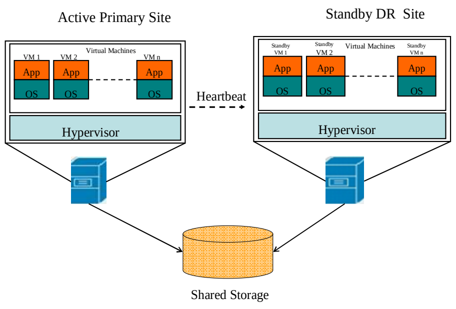

Figure 1. Active-Standby virtualized clustering architecture

We also consider the preventive maintenance for virtual machine servers in the (VLDR) plan. Software failures can occur at the operating system level and application level due to aging problem. In virtualization environments, the hypervisor itself or virtual machines can fail with software failure. To counteract software failure due to software aging problem in VMs, we employ the two kinds of preventive maintenance (software rejuvenation methodology) will be performed on the system is based on unstable state: a minimal maintenance (a partial system clean up) and a major maintenance (clean and restart). The VMs at primary site provide different services. During the minimal maintenance, an active VM can provide continued services because this maintenance is a partial system clean up. When one of the active VMs at primary site needs to do major maintenance, the services are migrated to one of the standby VMs at DR site.

VMs can be failed over through migration or restarted from storage onto standby server at DR site but the migration downtime is nearly zero and we can neglect the duration of migration. By using virtualization technology, it can provide continued services even if VMs need to perform major maintenance.

3.1. Modeling and Analysis of Proposed PM model for (VLDR)Plan

Petri Nets (PN) are a graphical tool for the formal description of the flow of activities in complex systems. PN used for modeling real systems are sometimes referred to as Condition/Events nets. Petri nets are extended by associating time with the firing of transitions, resulting in Timed Petri nets. A special case of Timed Petri nets is Stochastic Petri nets (SPN) where the firing times are consider to be random variables.

Using a stochastic Petri net model, we describe the behavior of preventive maintenance (PM) model for virtualized local disaster recovery plan in Figure 2. Each physical server includes the following transitions. The circles represent places and n represents the tokens that held inside that place. The robust and healthy state is modeled by the place $P_{H,O}$. It has n tokens which represented n virtual machines in active physical server. When transition T_I fires, the VM enters the inspection state and a token moves from $P_{H,O}$ to $P_{H,I}$. After inspection is complete (firing the transition T_{H},) no action is taken if the system is found to be in healthy state. Transition T_U models the unstable state of the VM. When this transition fires, (i.e., a token reaches place $P_{U,O}$) the VM is operational but in the unstable state. The transition $T_{U,I}$ models the unstable state and under inspection state of the VM. During the VM in the first unstable and minimal maintenance state, the transition $T_{U,M1}$ is enabled. After minimal maintenance is complete (firing the transition $T_{M1,H}$), the VM enters the healthy and operational state.

The transition T_{U2} models that the VM operational but in major unstable state. Once its fires, a token moves in the place $P_{U2,M}$ and the activity related with operational and major maintenance state. The transition $T_{M\text{-}SW}$ models that services of VM are migrated to the standby VM at DR site. After firing the transition $T_{M,H}$, the VM enters the healthy and operational state. If the VM had reached the failure state (token in P_F).

In the P_F state, failure can cause by hardware faults and when the transition T_{SW} fires the VMs are moved to standby physical host at DR site and takes overall the running operation of the fail physical server from primary site. Transition T_{SW} can only fire when the DR site available and a token moves from PF to P_{SW}. After the primary site come back online, the DR site will put back all operations it obtained from primary site. This return action is represented by the firing of T_{SWBK}. Similarly, the DR site physical node can be inspection transition, minimal maintenance and major maintenance action, repair transitions, switchover and switch back transitions.

Unless services are migrated, the transition T_{Down} will fire and reach the fail state. After that T_{UP} fires to repair the both physical hosts and return to healthy and operational state.

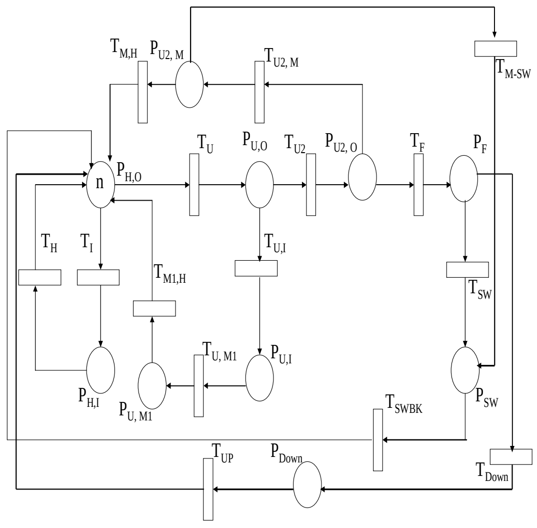

Figure 2. Stochastic Petri net model of PM model for (VLDR) plan

Places		**Descriptions**
$P_{H,O}$	:	VMs are healthy and operational state
$P_{H,I}$	:	VMs are under inspection state

$P_{U,O}$	:	VMs are unstable but it is operational state
$P_{U,I}$	:	VMs are in unstable and is under inspection state
$P_{U2,0}$	:	VMs are major unstable but is operational state
$P_{U, M1}$	:	VMs are in unstable and under minimal maintenance state
$P_{U2, M}$	:	VMs are in unstable and under major maintenance state
P_F	:	Physical host or VMs are in failure state
P_{SW}	:	VMs are switch to standby physic-cal server at DR site
P_{SWBK}	:	VMs are switch back from DR site when the primary host come back online
P_{Down}	:	Both physical hosts are in failure state

The assuming is that time to VM failure is hypo-exponential and the time to repair is exponentially distributed. Let λ_{in} be the inspection transition firing rate associated with T_I and $T_{U,I}$, and μ_{in} be the transition firing rate associated with $T_{U,M1}$, $T_{U2,M}$ and T_H. Further assuming that, an inspection is triggered after a mean duration $1/\lambda_{in}$.

We are interested in testing how two kinds of preventive maintenance can improve availability and lower downtime in virtualized local disaster recovery plan. To analyze the proposed model, we need to choose the firing rates for all transitions. The exact model firing rates of transitions for the model are not known, a good estimate value for a range of model firing rates is assumed. For this purpose, the transition firing rates which are needed to test the proposed model are used from literature review in references [7] and [13] as shown in Table 1.

Table 1. Transitions Firing Rates

Transition	**Firing Rate (hr^{-1})**
T_U , T_{U2} , T_F	1time/ a day
$T_{U,M1}$, $T_{U2,M}$,T_H	0.3
T_{UP}	1
$T_{M,H}$	2
$T_{M1,H}$	3
$1/T_{SW}$, $1/T_{M1\text{-}SW}$, $1/T_{M\text{-}SW}$	10 min
T_{SWBK}	2 times/a day
T_{Down}	1 time /3 days

3.2. Availability and Downtime in Analysis

In the proposed preventive maintenance model for (VLDR) plan, services are not available when both primary site and DR site are down.

We also define the availability of the proposed model as:

Availability = 1- Unavailability (1)

Availability = 1- P_{Down} (2)

where P_{Down} = The probability of the place P_{Down}

The expected total downtime with preventive maintenance in an interval of T time units is

Downtime (T) = (P_{Down}) x T (3)

3.3. Simulation Results

We describe the simulation results of the proposed model through SHARPE tool.

In Figure 3, we plotted the availability as a function of the mean time between inspections MTBI for 2 VMs. We use several different values of time to carry out the inspection. The availability reaches the maximum at MTBI= 10 and μ_{in} = 0.6.

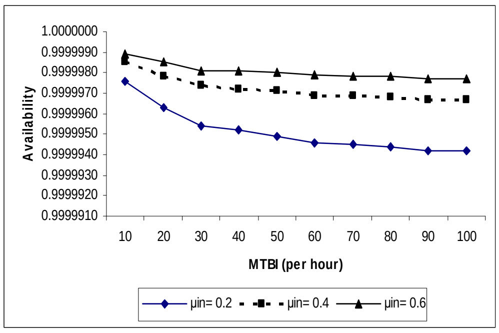

Figure 3. Availability vs MTBI for PM model with different μ_{in}

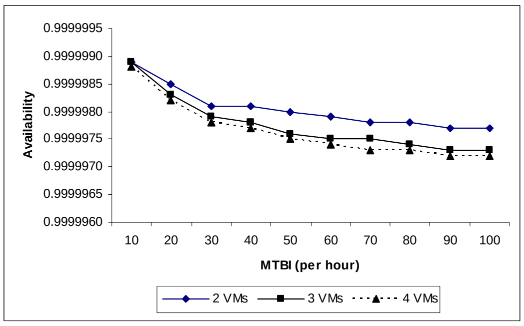

Figure 5. Availability vs MTBI for PM model with different VMs

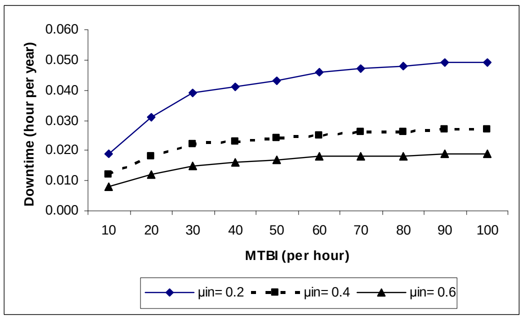

Figure 4. Downtime vs MTBI for PM model with different μ_{in}

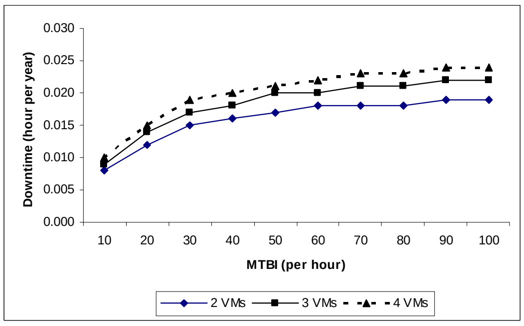

Figure 6. Downtime vs MTBI for PM model with different VMs

The downtime of the PM model for (VLDR) plan is shown in Figure 4. We plotted the downtime as a function of mean time between inspections MTBI with different μ_{in}. The downtime reaches the minimum at MTBI =10 and μ_{in}=0.6.

The change in availability of proposed PM model with different numbers of VMs and different mean time between inspections MTBI is plotted in Figure 5. According to the figure, the availability reaches the maximum at MTBI= 10 and μ_{in} = 0.6 for any number of VMs. There will be slightly different the availability changes among 2 VMs, 3 VMs and 4 VMs.

The downtime of the PM model for (VLDR) plan is shown in Figure 6. The downtime reaches the minimum at MTBI =10 and μ_{in}=0.6 for any number of VMs. According to the figure, the downtime (hour per year) is not very significant among 2 VMs, 3VMs and 4 VMs.

4. Conclusion

In this paper, we have presented the preventive maintenance model for virtualized local disaster recovery plan using a stochastic Petri net model in which two kinds of preventive maintenance are performed. We employed active-standby clustering architecture in the proposed model. We also analyze availability and downtime in hours per year, in terms of the firing rates of transitions in the model with evaluation through SHARPE tool. The obtained results showed that the proposed two kinds of preventive maintenance can helpful the virtualized local disaster recovery plan.

References

[1] A. Cohen, “Active-Active Virtualization: A More Cost Effective Approach to Cross Campus Disaster Recovery”, Sanrad, 2008.

[2] B. C. Martin, “Disaster Recovery Plan Strategies and Processes”, Version 1.3, February 2002.

[3] D. Clitherow, M. Brookbanks, N. Clayton, and G. Spear, ‘‘Combining High Availability and Disaster Recovery Solutions for Critical IT Environments’’, IBM Systems, *Journal* 47, No. 4, 563–575 (2008).

[4] E.Vargas, “High Availability Fundamentals”, Sun Blueprints Series, 2000.

[5] F. Salfner and K. Wolter, “A Petri Net Model for Service Availability in Redundancy Computing Systems”, *Proc. The 2009 Winter Simulation Conference, IEEE,* 2009.

[6] http://www.adams.edu.

[7] http://www.software-rejuvenation.com.

[8] http://www.veritas.com.

[9] J. L. Peterson, "*Petri Net Theory and the Modeling of Systems"*, Prentice-Hall, Englewood Cliffs, NJ, U.S.A.1981.

[10] K. S.Trivedi, "*Probability and Statistics with Reliability, Queuing, and Computer Science Applications”*, John Wiley and Sons, 2002.

[11] S. Garg, A. Puliafito , M. Telek, K. Trivedi, "Analysis of Preventive Maintenance in Transactions Based Software Systems", *IEEE Transaction on Computers*, VOL. 47, NO. 1, January, 1998, pp.96-107.

[12] X. Du, D. Hou, X. Zhong, Y. Qi, Y. Chen. "Modeling and Performance Analysis of Software Rejuvenation Policies for Multiple Degradation Systems", *The 33rd Annual IEEE International Computer Software and Applications Conference, 2009.*

[13] Y. Huang, C. Kintala, N. Kolettis and N. D. Fultion, "Software Rejuvenation: Analysis, Module and Applications", Proc. *The 25th International Symposium., Fault-Tolerant Computing, Pasadena, CA, June.1995*,pp.381-90.doi:10.1109/FTCS.1995.466961.