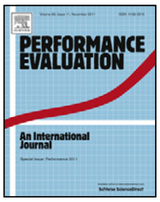
Performance Evaluation 70 (2013) 212–230

Contents lists available at SciVerse ScienceDirect

Performance Evaluation

journal homepage: www.elsevier.com/locate/peva



Modeling and analysis of software rejuvenation in a server virtualized system with live VM migration

Fumio Machida[a,*], Dong Seong Kim[b], Kishor S. Trivedi[c]

[a] *Knowledge Discovery Research Laboratories, NEC Corporation, Kawasaki, Japan*
[b] *Department of Computer Science and Software Engineering, University of Canterbury, Christchurch, New Zealand*
[c] *Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708, United States*

ARTICLE INFO

Article history:
Available online 18 September 2012

Keywords:
Analytic model
Live VM migration
Server virtualization
Software rejuvenation
Stochastic reward net

ABSTRACT

As server virtualization is used in a number of IT systems, the unavailability of virtual machines (VM) on server virtualized systems becomes a significant concern. Software rejuvenation is a promising technique for improving the availability of server virtualized systems as it can postpone or prevent failures caused by software aging in both the VM and the underlying virtual machine monitor (VMM). In this paper, we study the effectiveness of a combination of VMM rejuvenation and live VM migration. When a VMM needs to be rejuvenated, the hosted VMs running on the VMM can be moved to another host using live VM migration and continue the execution even during the VMM rejuvenation. We call this technique *Migrate-VM rejuvenation* and construct an availability model in the stochastic reward net for evaluating it in comparison with the conventional approaches; Cold-VM rejuvenation and Warm-VM rejuvenation. The designed model enables us to find the optimum combinations of rejuvenation trigger intervals that maximize the availability of VM. In terms of the maximum VM availability, Migrate-VM rejuvenation is potentially the best approach. However, the advantage of Migrate-VM rejuvenation depends on the type of live VM migration (stop-and-copy or pre-copy) and the policy for migration back to the original host after VMM rejuvenation (return-back or stay-on). Through numerical examples, we show that "pre-copy" live VM migration is encouraged rather than pure "stop-and-copy" migration and it is better to return back VM to the original host soon after the VMM rejuvenation (i.e., "return-back" rather than "stay-on" policy) for high-availability. The effect of the VMM rejuvenation technique on the expected number of transactions lost is also studied by combining the availability model with an $M/M/1/n$ queueing model.

1. Introduction

Server virtualization is an essential aspect of system infrastructures for various software services. Many organizations employ server virtualization to consolidate server infrastructure for their information services to reduce the total cost of ownership. Server virtualization is also used as an essential building block for cloud computing services that provide computing resources and services to users over the Internet or private networks. The users of cloud computing services can easily request and use virtual machines (VMs) through the provided user interfaces whenever necessary. Accordingly, high-availability assurance of VMs running on server virtualized systems becomes a common concern to many organizations, service providers and service users.

* Corresponding author. Tel.: +81 44 431 7032; fax: +81 44 431 7680.
E-mail addresses: f-machida@ab.jp.nec.com (F. Machida), dongseong.kim@canterbury.ac.nz (D.S. Kim), kst@ee.duke.edu (K.S. Trivedi).

0166-5316/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.peva.2012.09.003

Software rejuvenation is one of the promising techniques for assuring high-availability of server virtualized systems [1,2]. Since server virtualization is usually implemented in software called hypervisor or virtual machine monitor (VMM), the system may suffer from the risk of software aging and related system failures. Software aging is caused by aging-related bugs that are known to be difficult to fix completely during software development and test phases [3,4]. To postpone or prevent system failures caused by software aging in VMM, software rejuvenation is applicable to VMM as well as VM. Software rejuvenation is a failure-prevention method that clears aging status by restarting or resetting the software execution environment [5]. Compared to the traditional software rejuvenation used for common software components, VMM rejuvenation may need additional consideration to the downtime of hosted VMs running on a VMM because they might become unavailable during the VMM rejuvenation. Such downtime needs to be taken into account in determining an appropriate rejuvenation technique and in triggering VMM rejuvenation.

In our previous study [2], comprehensive availability models for analyzing the availability of server virtualized systems with time-based rejuvenation were presented. The designed models captured software aging states of VM and VMM as well as their failure states caused by software aging. We made comparisons of three different rejuvenation techniques for VMM, namely Cold-VM rejuvenation, Warm-VM rejuvenation, and Migrate-VM rejuvenation. The difference between the three techniques is based on the method used to deal with the hosted VM during VMM rejuvenation. Cold-VM rejuvenation simply shuts down the hosted VMs before triggering the VMM rejuvenation and restarts the VMs after the completion of VMM rejuvenation. Warm-VM rejuvenation suspends VMs before triggering the VMM rejuvenation and resumes the executions of the VMs after the completion of the VMM rejuvenation. Migrate-VM rejuvenation provides a judicious combination of VMM rejuvenation and live VM migration. Running VMs are first transferred to another host as long as the host is available and then VMM rejuvenation starts. During the VMM rejuvenation the VMs can continue their executions on the other host. Although the model for Migrate-VM rejuvenation in our previous paper was constructed simply in an analogous way to Warm-VM rejuvenation, our preliminary numerical solution revealed that Migrate-VM rejuvenation potentially outperforms the other two rejuvenation techniques in terms of the steady-state availability of VM as long as the downtime caused by the live migration is short enough.

In this paper, the effectiveness of Migrate-VM rejuvenation is further investigated by modeling it more precisely. The major extensions from our previous work [2] are as follows. First, the availability model for Migrate-VM rejuvenation is further improved by incorporating (i) failure of the migration target host due to software aging of the VMM on the host, (ii) failures of VM migration process, (iii) types of live VM migration (“stop-and-copy” or “pre-copy”), and (iv) the policies concerning VM migration to return back to the original host after VMM rejuvenation (“return-back” or “stay-on”). We divide the Migrate-VM rejuvenation into four classes according to the combinations of the types of migration and the stay-on or return-back policies. For each class, numerical solution is carried out to find the optimum rejuvenation schedules for VM and VMM that maximize the steady-state availability. Taking into account failures of the live VM migration process, we analyze the sensitivity to the coverage of live VM migration and explore the relationship between the coverage and the policy for return-back VM migration. Secondly, in addition to the analysis of the steady-state availability of VM, the number of transactions lost due to VMM rejuvenation is formulated and computed by combining the virtualized system model with an $M/M/1/n$ queuing model. Based on our numerical examples, we conclude that Migrate-VM rejuvenation with the combination of pre-copy technique and the policy enforcing immediate VM return back to the original host after VMM rejuvenation is the best option.

The rest of the paper is organized as follows. In Section 2, we discuss the related work on software aging and rejuvenation in server virtualized systems. Section 3 introduces three different techniques for VMM rejuvenation. In particular, for Migrate-VM rejuvenation, types of live VM migration, failures of migration process and policies for VM migration are introduced. Section 4 describes the availability models for server virtualized systems with the different VMM rejuvenation techniques. We use the high-level formalism of stochastic reward nets (SRN) in order to simplify the construction of availability models. Section 5 provides the results of our numerical examples for the three VMM rejuvenation techniques and shows the effectiveness of Migrate-VM rejuvenation. Besides steady state availability, the expected number of transactions lost is computed by introducing an $M/M/1/n$ queuing model. The effects of live VM migration failure, the live VM migration types, and the VM return-back policies are investigated by sensitivity analysis. Finally Section 6 concludes the paper. A brief introduction of SRN is provided in the Appendix.

2. Related work

Server rejuvenation is often performed by rebooting the operating system which clears all of the internal states that may cause software aging [5]. Aging-related bugs in operating systems are reported in previous literature and are known to cause various adverse effects such as resource depletion, performance degradation and system failure [6–8]. Server rejuvenation is used in the cluster environment as well and the rejuvenation triggers can be controlled by a commercial tool [9]. Since many VMMs are implemented by extending operating systems, they might suffer from similar aging-related bugs and associated failures.

Software rejuvenation of VMMs for server virtualization was first studied in [10]. If software rejuvenation is applied to VMMs directly, all the hosted VMs go down as well because of the dependency on the VMM (i.e., hypervisor). In [10] the authors presented Warm-VM reboot as a new technique for VMM rejuvenation that suspends the execution of hosted VMs in memory before starting the VMM rejuvenation and resumes their execution after the completion of the rejuvenation. The