

Model-based Survivability Analysis of a Virtualized System

Xiaolin Chang^a, Zhenjiang Zhang^b, Xiaodan Li^c, Kishor S. Trivedi^c

^aSchool of Computer and Information Technology, Beijing Jiaotong University, P.R. China, xlchang@bjtu.edu.cn

^bSchool of Electronic and Information Engineering, Beijing Jiaotong University, P.R. China, zhjzhang1@bjtu.edu.cn

^cDepartment of Electrical and Computer Engineering, Duke University, Durham, NC, {xiaodan.li, ktrivedi}@duke.edu

Abstract—Transient survivability analysis of a virtualized system (VS) is critical to the wide deployment of cloud services. The existing research of VS availability and/or reliability focused on the steady-state analysis. This paper presents a model and the closed-form solutions to analyze the survivability of both cloud service and VS after a service breakdown occurrence by using continuous-time Markov chain. Service breakdown may be caused by software rejuvenation of virtual machine (VM) and/or VM monitor (VMM), or caused by VM and/or VMM bugs. The VS applies two techniques for improving service survivability: VM failover and live VM migration. The proposed model and the defined survivability metrics not only enable us to quantitatively assess the system survivability but also provide insights on the investment efforts in system recovery strategies. Sensitivity analysis through numerical analysis is carried out to study the impact of key parameters on system survivability.

Index Terms— Live VM migration; Markov chains; Survivability; Transient analysis; Virtualized system

I. INTRODUCTION

Large downtime of cloud service may lead to productivity loss and even business loss. The common high availability (HA) techniques adopted in a virtualized system include software rejuvenation [1], failover and live VM migration [2]. The tremendous growth in virtualized system deployment gives rise to the need for performance analysis of the transient behaviors of a virtualized system with these techniques. Survivability is a transient measure and is generally defined as the ability of the system to recover predefined service in a timely manner after the occurrence of disaster [3]. Survivability analysis could help improve the systems' capability to provide critical services when damage occurs to part of the system. A disaster could be any kind of undesired occurrence to the system of interest. For example, cloud service breakdown.

This paper targets the survivability assessment of the cloud service and virtualized system after a service breakdown occurs. The breakdown may be caused by virtual machine (VM) rejuvenation, virtual machine monitor (VMM) rejuvenation, VM bugs, and/or VMM bugs. We quantify the survivability as the transient performance of the virtualized system from the instant when the service breakdown occurs until the full system recovers. By full system recovery, we mean that not only the cloud service is in normal state but also the damaged parts of the system complete their repair. The system of interest consists of three main components: *Main host*, *Backup host* and *Management host*, shown in Figure 1. There are more than one VM in *Main host*: active VM and standby/backup VMs. There are standby VMs in *Backup host*. Cloud service, denoted as App in the following, is

deployed in the active VM. The virtualized system applies two techniques for improving service survivability: VM failover and live VM migration. When the active VM fails, a standby VM on the same host is used or it is migrated to the other host for continuing cloud service. The unique features of system virtualization highlight the challenges to address the survivability for the virtualized system.

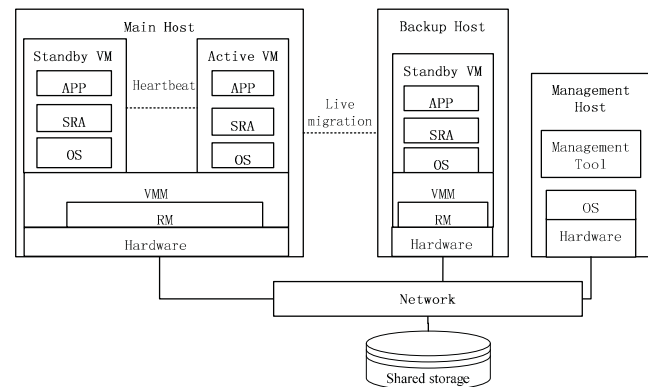


Figure 1 The System Architecture

We present a survivability model by employing continuous-time Markov chain (CTMC) with rewards. Survivability metrics of interest are furthermore defined and the closed-form solutions to these metrics are derived in order to quantify the attributes of both cloud service and virtualized system during the virtualized system recovery. The attributes include (i) the downtime of cloud service by time t ; (ii) the probabilities of the cloud service and virtualized system recovery by time t ; (iii) mean time to recover cloud service; and (iv) mean time to recover the virtualized system. We also carry out sensitivity analysis through numerical experiments.

Recently, studies have been carried out for the cloud service availability analysis by adopting analytic modeling approach. Most focused on steady-state analysis, see [8] and references therein. Although survivability analysis has been conducted in the other fields [4][5], there are few researchers to analyze the survivability of the virtualized environment. To the best of our knowledge, no work exists that analyzes and evaluates the survivability analysis of the virtualized system considered in this paper. In [6], the authors analyzed the survivability of three VM-based architectures through simulations. Their survivability model is single phase. But our model is a multiple-phase. When there are multiple actions in the recovery process, a multiple-phase recovery model could more effectively capture the

characteristics of the restoration process, compared to merging into single phase. In [7], the authors did not consider the different sources causing service breakdown and did not consider the recovery of the whole data centers.

The paper is organized as follows. Section II presents the system model and model solutions. In Section III, we present evaluation results. The conclusion is drawn in Section IV.

II. SYSTEM DESCRIPTION AND MODEL

This section first presents an overview of the virtualized system of interest in this paper. Then the CTMC model is presented.

A. The System Architecture

The virtualized system consists of three main components: *Main host*, *Backup host* and *Management host*. See Figure 1. TABLE I defines the parameters to be used in the rest of the paper.

TABLE I. PARAMETER DEFINITION

Notation	Definition	Value in default
p_1	Probability that service breakdown is due to the active VM rejuvenation or its aging-related failure occurrence.	0.7
p_2	Probability that service breakdown is caused by VM crashes.	0.1
p_3	Probability that service breakdown is due to the active VMM rejuvenation or its aging-related failure occurrence.	0.1
p_4	Probability that VMM crashes	0.1
$1/\gamma$	Decision time.	Very large
$1/\beta_0$	Mean VM switching time due to VM rejuvenation	3.0 seconds
$1/\beta_1$	Mean VM switching time due to VM crash	$2 * \beta_0$
$1/\beta_2$	Mean VM reboot time	60 seconds
$1/\beta_3$	Mean VM repair time	1 hour
$1/\beta_4$	Mean VM migration time due to VMM rejuvenation	60 seconds
$1/\beta_5$	Mean VM migration time due to VMM crash	$2 * \beta_4$
$1/\beta_6$	Mean VMM restart time	180 seconds
$1/\beta_7$	Mean Host repair time	24 hours

Main host contains a VMM, which runs an active VM with a desired application and several standby VMs. *Backup host* is a spare host which could perform the *Main host* role when a VM migrates to it successfully. It also has several standby VMs, which could be used by failover mechanism after this *Backup host* takes the role of *Main host*. *Management host* is a component responsible for controlling the entire cloud environment to detect the VMM failure by means of a specific cloud management tool. We assume that VM images and VMM code files are stored in shared storage (SS). In addition, all the running information of the active VMs is stored in SS periodically. Then these data can be used for failover, VM restart, VMM restart and live VM migration. An agent is installed in each VM for the VM rejuvenation operation. Similarly, an agent is deployed in the VMM in order to analyze all VMs running on this VMM, detect their abnormal behaviors and trigger the rejuvenation of that VM when it detects anomalies of a VM.

All VMs (including primary and standby) and VMMs of both *Main* and *Backup hosts* suffer from a non-aging Mandelbug-related failure and aging-related failure. We assume that at least a standby VM on a VMM is available at any time. The active VM's running state stored in the NAS could be sent to the selected standby VM. When the active VM fails due to VM rejuvenation or aging-related bug or non-aging Mandelbug-, the App deployed in the active VM is broken. Then one standby VM is selected to takes charge of the App and then acts as an active VM during its repair period. A failed VM will turn to be a standby VM after repair, rejuvenation or recovery. We assume that the physical machine and SS never fail. When VM crashes, *Management host* needs some time to make failover decision. Thus, β_1 is larger than β_0 .

Live VM migration is used to migrate the active VM to *Backup host* when the VMM needs to be rejuvenated or crashes. This technique moves the active VM with all the requests and sessions from *Main host* to *Backup host* such that there is no loss in any in-flight request or session data during the rejuvenation or repair. We assume the migration never fails. When VMM crashes, the VM information stored in NAS are migrated to *Backup host* but *Management host* needs some time to make migration decision. Thus, β_5 is larger than β_4 .

There exists dependency between some components. That is, the change of a component state triggers the state change of its dependent component. We define $A \succ B$ as that A depends on B. Thus, in the virtualized system, the component dependency is $\text{App} \succ \text{VM} \succ \text{VMM}$. We assume that there is no service breakdown before the virtualized system recovery from a service breakdown.

B. The Survivability Model

Figure 2 describes the survivability model, which is a Markov model with an absorbing state. This model does not include the failure process leading to service breakdown. The initial state is conditioned to be the state of service breakdown. The mean time to determine the cause of the service breakdown is denoted as $1/\gamma$. This mean time value is very smaller than the other intervals of time considered in this paper and then we assume $\gamma = \infty$. Therefore, the average time spent in state "Service breakdown" is zero. The initial probabilities for state 1,2,3 and 4 are then equal to p_1, p_2, p_3 and p_4 .

We assume that no other failure occurs during the system recovery. All the states except the absorbing state are transient and do not have a steady state probability. The definition of each state is given in TABLE II. When the cloud service breakdown occurs due to the active VM rejuvenation or the active VM reboot caused by an aging-related bug in VM. A standby VM on the same host is selected for handling the application/cloud service. This failover process requires very small delay, denotes as $1/\beta_0$. Meanwhile, the VM must restart with time $1/\beta_2$. VMC represents that service breakdown is due to VM crash caused by non-aging Mandelbug in VM. The failover process requires a bit larger delay, denotes as $1/\beta_1$. Meanwhile, the VM must be

repaired with time $1/\beta_3$. After repair, the system goes to “VM ready to reboot” and it takes $1/\beta_2$ time to run cloud service.

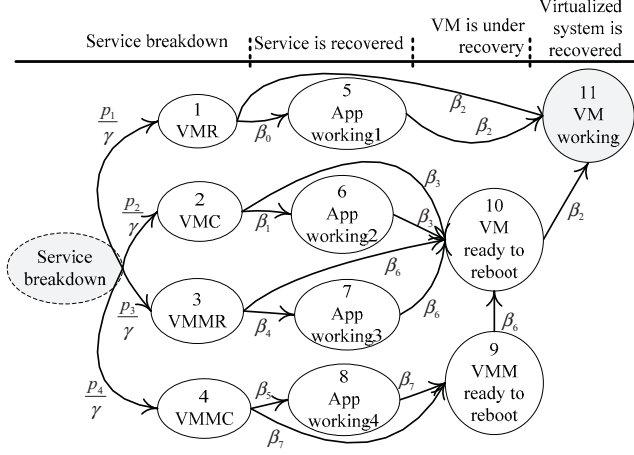


Figure 2 Phased recovery model of virtualized system

TABLE II. STATE NOTATIONS

Notation	Definition
VMR (1)	Service breakdown due to VM rejuvenation or VM aging-related failure
VMC (2)	Service breakdown due to VM non-aging Mandelbug-related failure
VMMR (3)	Service breakdown due to VMM rejuvenation
VMMC (4)	Service breakdown due to VMM non-aging Mandelbug-related failure
App working1 (5)	Cloud service is recovered from the breakdown, which is caused by the active VM rejuvenation or the active VM reboot caused by an aging-related bug in VM.
App working2 (6)	Cloud service is recovered from the breakdown, which is caused by the active VM crash.
App working3 (7)	Cloud service is recovered from the breakdown, which is caused by the active VMM rejuvenation or the VMM reboot caused by an aging-related bug in VM.
App working4 (8)	Cloud service is recovered from the breakdown, which is caused by the VMM crash.
VMM ready to reboot (9)	VMM software can work but needs to start.
VM ready to reboot (10)	VM could work but needs to start.
VM working (11)	VM could run applications.

VMMR denotes the situation where the cloud service breakdown occurs due to the VMM rejuvenation, namely the VMM reboot caused by an aging-related bug in VMM. Live VM migration is applied with time $1/\beta_5$. VMM restart time is denoted as $1/\beta_6$. VMMC represents that cloud service breakdown is caused by VMM crash caused by a non-aging Mandelbug in the VMM. The live VM migration process requires a bit larger delay, denotes as $1/\beta_5$. The crashed VMM is repaired with the repair time of $1/\beta_7$. After repair, the system goes to “VMM ready to reboot” and takes $1/\beta_7$ time to run VM.

Let $A = \beta_0 + \beta_2$, $B = \beta_1 + \beta_3$, $C = \beta_4 + \beta_6$ and $D = \beta_5 + \beta_7$. Let $\pi_i(t)$ be the transient probability that the system is in state i

at time t and then $\sum_{i=1}^{11} \pi_i(t) = 1$. The initial state probabilities of

each state are set as $\pi_1(0) = p_1$, $\pi_2(0) = p_2$, $\pi_3(0) = p_3$, $\pi_4(0) = p_4$ and $\pi_i(0) = 0$ for $i \in [5, 11]$. Then we can derive the closed-form expressions for $\pi_i(t)$ at time t : $\pi_1(t) = e^{-At} \pi_1(0)$, $\pi_2(t) = e^{-Bt} \pi_2(0)$, $\pi_3(t) = e^{-Ct} \pi_3(0)$, $\pi_4(t) = e^{-Dt} \pi_4(0)$, $\pi_5(t) = \pi_1(0)(e^{-\beta_2 t} - e^{-At})$, $\pi_6(t) = \pi_2(0)(e^{-\beta_3 t} - e^{-Bt})$, $\pi_7(t) = \pi_3(0)(e^{-\beta_6 t} - e^{-Ct})$, $\pi_8(t) = \pi_4(0)(e^{-\beta_7 t} - e^{-Dt})$,

$$\begin{aligned} \pi_9(t) &= \frac{\beta_7 \pi_4(0)}{\beta_7 - \beta_6} (e^{-\beta_6 t} - e^{-\beta_7 t}), \\ \pi_{10}(t) &= \frac{\beta_3 \pi_2(0)}{\beta_3 - \beta_2} (e^{-\beta_2 t} - e^{-\beta_3 t}) + \\ &\quad \frac{\beta_6 (\beta_7 - \beta_6) \pi_3(0) + \beta_7 \pi_4(0)}{(\beta_7 - \beta_6)(\beta_6 - \beta_2)} (e^{-\beta_2 t} - e^{-\beta_6 t}) - \\ &\quad \frac{\beta_7 \beta_6 \pi_4(0)}{(\beta_7 - \beta_6)(\beta_7 - \beta_2)} (e^{-\beta_2 t} - e^{-\beta_7 t}) \end{aligned}$$

and $\pi_{11}(t) = 1 - \sum_{i=1}^{10} \pi_i(t)$.

Associating each state with a reward rate allows the computation of the survivability metrics of interest. ϖ_i is defined to denote the average reward of state i . Then, when $\varpi_i = 1$ for $i \in [1, 4]$ and $\varpi_i = 0$ for $i \in [5, 11]$, $\sum_{i=1}^{11} \int_0^t \pi_i(x) dx$ is downtime of cloud service at time t . Similarly, we can compute the mean times for full APP recovery and full virtualized system recovery. $L_i(\tau) = \int_0^\tau \varpi_i \pi_i(x) dx$ is defined to denote the mean reward accumulated by time τ at state i after a service breakdown. Based on these formulas and definitions, we get the computation expressions for the following metrics:

- 1) Probability that App is recovered by time t is $\sum_{i=5}^{11} \pi_i(t)$.
- 2) Probability that the virtualized system is recovered by time t is $\pi_{11}(t)$.
- 3) Mean time up to full APP recovery is $\int_0^\infty \sum_{i=1}^4 L_i(x) dx$.
- 4) Mean time to full virtualized system recovery is $\int_0^\infty \sum_{i=1}^{10} L_i(x) dx$.

III. NUMERICAL ANALYSIS AND DISCUSSION

This section evaluates our model solutions. All time intervals in the model are assumed to be exponentially distributed. The model parameter values are set according to the existing related literature [8] and references therein. Unless otherwise specified, the parameter values are set as in TABLE I.

Figure 3 depicts the probability that App which has been recovered by time t by varying the initial probabilities of the first four states. “0.7_0.1_0.1_0.1” denotes the results under $p_1 = 0.7$

and $p_2 = p_3 = p_4 = 0.1$. Similarly, “0.1_0.7_0.1_0.1” denotes the results under $p_2 = 0.7$ and $p_1 = p_3 = p_4 = 0.1$. We observe that the accumulated recovering probability of App increases over time. We also observe that App can recover quickly in case of $p_1 = 0.7$, denoting that the service breakdown is caused mainly by VM aging-related bugs/VM rejuvenation. The corresponding recovery probability reaches 0.9 in less than 60 seconds. However, when the service breakdown is majorly due to VMM crash (that is, 0.1_0.1_0.1_0.7), it takes much more time for App to recover, compare to VM failure. This is demonstrated by the results of 0.1_0.1_0.1_0.7. We could see that App recovery probability has not reached 0.9 after 200 seconds. The slow increase in the App recovery probability leads to the larger service downtime.

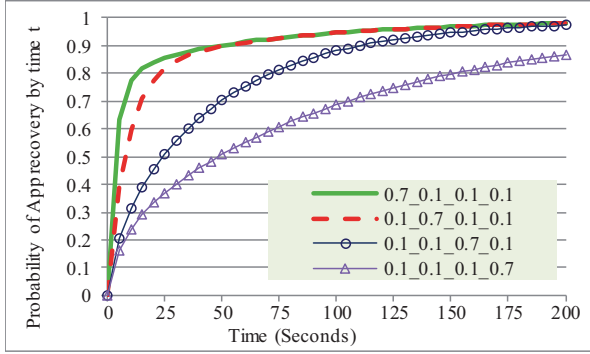


Figure 3 Probability of App being recovered by time t by varying p_1 , p_2 , p_3 and p_4

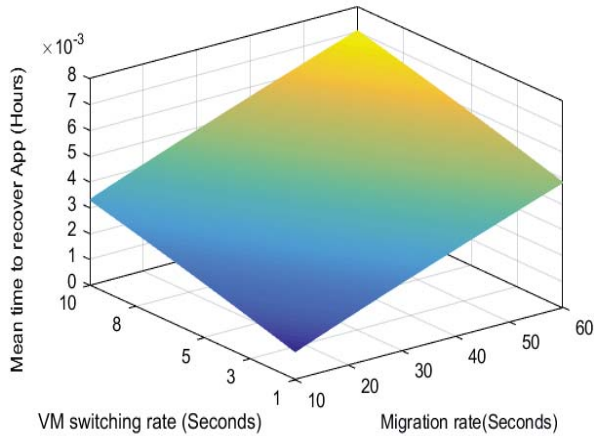


Figure 4 Mean time to recover App over VM switching and migrating rates

Figure 4 shows the mean time to App recovery under different VM switching rates and live VM migration rates. VM switching rate β_0 varies from 1 second to 10 seconds. VM migration rate varies from 10 seconds to 60 seconds. The result of Figure 4 could help determine the combination of VM switching and migration rates to optimize the revenue of service provider. Figure 5 shows the mean time to full virtualized system recovery under different failover rates and live VM migration rates. VM repair rate varies from 0.5 hours to 3 hours. VMM repair

rate varies from 3 hours to 24 hours. Similarly, the results of Figure 5 could help derive the combination of VM and VMM repair rates to obtain the maximum revenue of service provider.

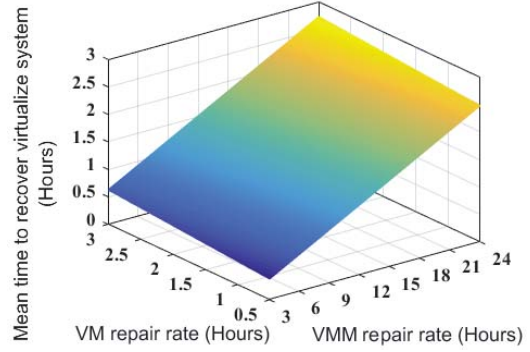


Figure 5 Mean time to recover virtualized system over VM and VMM repair rates.

IV. CONCLUSION AND FUTURE WORK

Virtualized systems have been applied to provision cloud service. Survivability issues must be addressed before this technique could be used for crucial services. This paper explores the CTMC model-based survivability analysis of a virtualized system after a cloud service breakdown occurs. We propose the model and define the survivability metrics. Numerical analysis is carried out to study the impact of the underlying parameters on the system survivability. These results could provide insights on the cost/benefit trade-offs of investment efforts in system recovery strategies.

ACKNOWLEDGMENT

The research of Xiaolin Chang is supported in part by NSF 61572066 of China. The research of Xiaodan Li and Kishor S. Trivedi is supported in part under US NSF grant number CNS-1523994, by IBM under a faculty grant and by NATO under Science for Peace project number 984425.

REFERENCES

- [1] Kishor S. Trivedi, Kalyanaraman Vaidyanathan: Software Rejuvenation - Modeling and Analysis. IFIP Congress Tutorials 2004: 151-182.
- [2] C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, “Live Migration of Virtual Machines,” NSDI 2005.
- [3] K.S. Trivedi, and R.F. Xia, “Quantification of system survivability,” *Telcommunication Systems* 60(4): 451-470, 2015.
- [4] Y. Liu, and K.S. Trivedi, “A General Framework for Network Survivability Quantification,” *MMB* 2004: 369-378.
- [5] P.E. Heegaard, K.S. Trivedi, “Network survivability modeling,” *Computer Networks* 53(8): 1215-1234, 2009.
- [6] J.J. Zheng, H. Okamura, and T. Dohi, “Survivability Analysis of VM-Based Intrusion Tolerant Systems,” *IEICE Transactions* 98-D(12): 2082-2090, 2015.
- [7] B. Silvaa, P. Maciela, A. Zimmermann and J. Brilhantea, “Survivability Evaluation of Disaster Tolerant Cloud Computing Systems,” *Proc. Probabilistic Safety Assessment & Management conference* 2014.
- [8] T.A. Nguyen, D.S. Kim, and J.S. Park, “Availability modeling and analysis of a data center for disaster tolerance,” *Future Generation of Computer Systems*, 27-50, 2016.