Availability Analysis of Cloud Deployed Applications

Manar Jammal
ECE Department
Western University
London ON, Canada
mjammal@uwo.ca

Ali Kanso
Ericsson Research
Ericsson
Montreal, Canada
ali.kanso@ericsson.com

Parisa Heidari
Ericsson Research
Ericsson
Montreal, Canada
parisa.heidari@ericsson.com

Abdallah Shami
ECE Department
Western University
London ON, Canada
ashami2@uwo.ca

***Abstract*—High availability (HA) is a main key performance indicator for cloud deployed services. Cloud providers offer different availability zones possibly located in different geographical regions. To protect cloud services against failures and natural disasters, it is recommended to deploy the applications on redundant resources across multiple zones and distribute the workload through a load-balancer. Different cloud infrastructure, located in different geographical zones with different energy source powering, hardware quality, etc., may have different reliability levels. Scheduling a cloud service on different zones while meeting the service level agreement availability requirements necessitate a solution to assess the expected availability of a given deployment. To quantify the expected availability offered by an application deployment, a formal stochastic model is required to capture the stochastic behavior of failures. This paper proposes a stochastic Petri Net approach that captures the stochastic characteristics of cloud services and translates them into elements of an availability model. This approach evaluates the availability of cloud services and their deployments in geographically distributed data centers (DCs). The results are useful to generate guidelines for an HA-aware scheduling.**

***Index Terms*—High availability, cloud applications, software components, stochastic failures, stochastic Petri Nets, recovery.**

I. Introduction

With the proliferation of on-demand cloud services that are expected to be available anywhere and anytime, service availability is an important requirement. Availability is defined as the percentage of time where these services are available in a given duration. It is important to assess the expected availability of a given deployment for both the cloud tenants and providers that are bound by a service level agreement. Different types of hardware and software failures can happen and cause service outage. These failures have a stochastic nature. Cloud users cannot prevent these failures' happenings. Some of the cloud users have their own proprietary High Availability (HA) solution to mitigate the service downtime [1]. An HA evaluation model is required to identify failures, their underlying causes, and attenuate associated risks and service outages. Stochastic Petri Nets (SPNs) and Markov chains are the approaches already used in the reliability/availability analysis of many complicated information technology (IT) systems [2] [3]. A comprehensive and analytical model for availability analysis is still required to capture the application behavior in a cloud setting.

The cloud model typically consists of multiple data centers each having a set of servers and a set of applications with multiple components. Using the appropriate scheduling solution, the applications are hosted on the servers that best fit the application requirements using VM (or containers) mapping. Consequently, any DC/server's failure mode can bring the hosted application down whether it is a planned or unplanned outage. Unplanned downtime can be defined as the time where a system enters a failure mode and becomes unavailable. Such downtime is a result of unexpected failure event and consequently neither the cloud provider nor the users are notified of it in advance. Therefore, it is necessary to have a model that takes into account the actual effect of failures on the system's availability. There are different forms of failures:

1) *Hardware/Infrastructure failures* [2] [3]: happening at the data center and server layers, they can be the results of faulty server's, storage's, and network's elements (e.g., faults in memory chips) and can be captured by the failure rates of the servers as well as the entire DC.
2) *Application failures* [4]: Such defects occur at the applications' and VMs' levels. They might be generated from the hypervisor malfunctioning, unresponsiveness of the operating system, files corruption, or viruses and software bugs, such as Heisenbugs, Bohrbugs, Schroedinbugs, or Mandelbugs [5]. Such failures are captured by the failure rate of the components and VMs.
3) *Force majeure failures* [6]: generated from power loss, storms, floods, and other natural disasters, these failures affect both the cloud provider infrastructure and the cloud applications. Due to their scale, we capture such failures by including them in the failure rate of the DC.
4) *Cascading failures*: being the results of an accumulated impact of hardware or software failure, can cease the functionality of DCs and the corresponding servers, VMs or applications (e.g., a dynamic host configuration protocol (DHCP) server malfunctioning can flood the network with DHCP requests causing a DC failure, followed by failure of the servers, and their hosted applications/VMs. Due to their propagation impact, we capture such failures by the failure rate of the DC.

Each of the previous failure states is associated with a failure

978-1-5090-1961-8/16 $31.00 © 2016 IEEE
DOI 10.1109/IC2E.2016.44

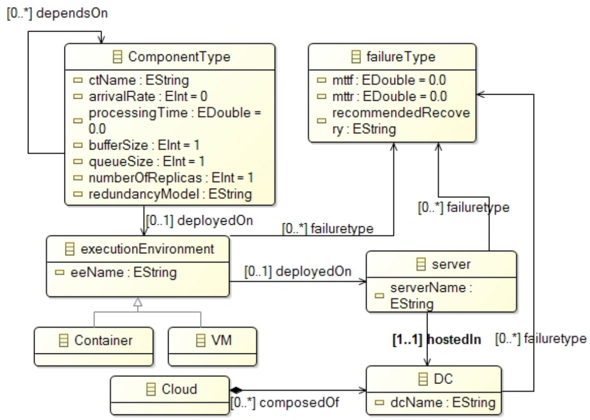

Fig. 1: Simplified UML model for a cloud deployment

rate or mean time to failure (MTTF) and mean time to repair or recover (MTTR) determined by the used repair or recovery policy. Due to the stochastic nature of the failures, we use probabilistic distribution functions such as exponential, Weibull, normal, or any other stochastic model to generate the failures. We consider a deterministic or a stochastic recovery depending on the used recovery or repair policy.

II. Approach

A typical cloud deployment is composed of multiple software components running on an execution environment (e.g., VM or container). The VM is hosted on a server, and the server in turn is hosted on a data center. Fig. 1 illustrates our simplified Unified Modeling Language (UML) model that captures such cloud deployment. To address the challenges of HA-aware scheduling discussed earlier, we need a behavioral model that can capture the stochastic behavior of the system (e.g., different failures) as well as its deterministic behavior (e.g., recovery actions). Stochastic Petri Nets are high level formal models to perform stochastic analysis and simulate the behavior of systems with stochastic behavior. To model the behavior of an application running on cloud, we have used Stochastic Colored Petri Nets (SCPN) [7], that captures both stochastic and deterministic events. In SCPN, the tokens can have different colors (types). Our approach is based on mapping an instance of the UML model describing a given deployment of the application in the cloud to the corresponding SCPN model. The model is simulated and analyzed using a simulator tool TimeNet to quantify the expected availability of the application [8].

Creating the SCPN model manually can be a tedious, time consuming and error prone task. To mitigate this complexity, we have defined a one to one mapping to achieve the transformation from the UML model of a cloud system to the corresponding SCPN model. This way, we have automated the model transformation from a UML model to the SCPN model which, will be analyzed with TimeNet tool. The exponential failure distribution has been used in many previous failure analysis and availability related work: [9], [10], [11], [12], and [13]. In this paper also, we use the exponential failure distribution to reflect failure rate or MTTF of DCs, servers, and applications/VMs. Such distribution is applied on all the stochastic failure transitions of the proposed SCPN model. The repair/recovery timed transitions, are modeled using a deterministic distribution. Note that our approach also supports other failure rates, as our model does not depend on a specific probability distribution.

III. Conclusion

Cloud services may become unavailable due to various stochastic failures while they are expected to be accessible at anytime and anywhere. In this paper, we explained an automated approach to capture the stochastic behavior of cloud systems and assess availability aspects of such systems. The paper proposed a SCPN approach that evaluates the availability of cloud services and their deployments in inter- or intra-data centers. This approach considers different failure types, functionality constraints, redundancy models, and interdependencies between different components' applications.

Acknowledgements

This work is partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC-STPGP 447230) and Ericsson Research.

References

[1] NETFLIX, "AWS Re:Invent - High Availability Architecture at Netflix," http://www.slideshare.net/adrianco/high-availability-architecture-at-netflix, December 2012.

[2] D. S. Kim, F. Machida, and K. S. Trivedi, "Availability modeling and analysis of a virtualized system," *15th IEEE Pacific Rim International Symposium on Dependable Computing,* pp. 365-371, 2009.

[3] W. E. Smith, K. S. Trivedi, L. A. Tomek, and J. Ackaret, "Availability analysis of blade server systems," *IBM Systems Journal,* vol. 47, no. 4, pp. 621-640, 2008.

[4] M. Grottke, A. P. Nikora, and K. S. Trivedi, "An empirical investigation of fault types in space mission system software," *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN),* pp. 447-456, 2010.

[5] K. Ramo, "Eliminating Software Failures A Literature Survey," *Licentiate Thesis,* 2009, http://www.doria.fi/bitstream/handle/10024/61561/nbnfi-fe201005051790.pdf?sequence=3.

[6] P. Bodik, F. Armando, M.J. Franklin, M.I. Jordan, and D.A. Patterson, "Characterizing, modeling, and generating workload spikes for stateful services," *ACM Symposium on Cloud Computing,* pp. 241-252, 2010.

[7] N. Gharbia, C. Dutheilletb, and M. Ioualalen, "Colored stochastic Petri nets for modelling and analysis of multiclass retrial systems," *Math. Comput. Model.,* vol. 49, no. 7-8, pp. 1436-1448, 2009.

[8] A. Zimmermann, "Modeling and Evaluation of Stochastic Petri Nets With TimeNET 4.1," *6th International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS),* pp. 54-63, 2012.

[9] F. Machida, D. S. Kim, and K. S. Trivedi, "Modeling and analysis of software rejuvenation in a server virtualized system," *2nd International Workshop on Software Aging and Rejuvenation,* pp. 1-6, 2010

[10] J. Xu, X. Li, Y. Zhong, and H. Zhang, "Availability modeling and analysis of a single-server virtualized system with rejuvenation," *Journal of Software,* vol. 9, no. 1, pp. 129-139, 2014.

[11] T. Thein and J. S. Park, "Availability analysis of application servers using software rejuvenation and virtualization," *Journal of Computer Science and Technology,* vol. 24, no. 2, pp. 339-346, 2009.

[12] M. Jammal, A. Kanso, and A. Shami, "High Availability-Aware Optimization Digest for Applications Deployment in Cloud," *IEEE International Conference on Communications (ICC),* pp. 6822-6828, 2015.

[13] M. Jammal, A. Kanso, and A. Shami, "CHASE: Component High-Availability Scheduler in Cloud Computing Environment," *IEEE International Conference on Cloud Computing (CLOUD),* pp. 477-484, 2015.