Automatic Testing Framework for Virtualization Environment

Chia Hung Kao
Cloud System Software Institute
Institute for Information Industry
Taipei, Taiwan
Email: chkao@iii.org.tw

Ping-Hsien Chi
Cloud System Software Institute
Institute for Information Industry
Taipei, Taiwan
Email: tailschi@iii.org.tw

Yi-Hsuan Lee
Cloud System Software Institute
Institute for Information Industry
Taipei, Taiwan
Email: erinlee@iii.org.tw

***Abstract*—Virtualization has attracted considerable attention in recent years. With the increasing popularity of virtualization technology, how to ensure the correctness and the quality becomes a critical issue. Several researches evaluate functionalities and corresponding performance on different virtualization environments. However, the task of testing and evaluation of virtualization environment is still complex and time consuming. In this paper, an automatic testing framework is introduced. The framework provides automatic methods of test environment setting, usage scenario deployment and test plan revision, and helps to facilitate the functional and performance evaluation on virtualization environments.**

I. Introduction

Virtualization has attracted considerable attention in recent years. Famous solutions such as KVM, VMware, Xen, Hyper-V, VirtualBox and so on, are widely used by enterprises, companies and individuals in their daily tasks. With the increasing popularity of virtualization technology, how to ensure the correctness and the quality becomes a critical issue.

Academia, industry and open source communities have proposed several testing and evaluation methods for virtualization environments [1] [2]. For example, Linux Virtualization Tests (virt-test) is a suite of tests made to exercise Linux virtualization hypervisors. SPECvirt and VMmark are performance benchmarks for different virtualization platforms. Several researches evaluate specific functionalities (e.g., live migration, fault tolerance and so on) and corresponding performance on different virtualization environments [3] [4]. However, the efficiency of testing and evaluation of virtualization environment is still hindered by several issues, which state as the following.

- Deployment and configuration of the test environment: The deployment and configuration of the test environment is complex and time consuming. For instance, the hardware specification, software environment and network setting of virtual environments should be taken into consideration before test execution. Automatic deployment and configuration can facilitate the testing and evaluation tasks.
- Deployment of usage scenarios: Usage scenarios should be taken into account when performing testing and evaluation tasks. For instance, migration during certain amount of accesses to the web server on virtual machine helps to validates the correctness by specific usage scenario. Similarly, automatic deployment of usage scenarios can facilitate the testing and evaluation tasks.
- Automatic adjustment of test plan: There are large amount of configurations and usage scenarios of virtualization environments. Therefore, automatic adjustment of test plan according to previous test result (e.g., increase load when previous result is pass) can decrease the configuration effort and discover potential defects efficiently.

In this paper, an automatic testing framework is introduced. The framework aims to provide automatic methods of test environment setting, usage scenario deployment and test plan revision. This helps to facilitate the functional and performance evaluation on virtualization environments.

II. Design of The Framework

As shown in Fig 1, the framework comprises several components, including Test Manager, Environment Manager, Test Controller, Log Collector, Test Case Repository, Result Repository, Configuration Script Repository and Test Script Repository. Test Manager is responsible for coordinating with other components to fulfill the testing tasks. It retrieves test cases from test case repository and parses related information (e.g., test steps and associated scripts) for Environment Manager and Test Controller to perform environment setting and testing. Necessary configuration scripts and test scripts are managed in the corresponding repositories. After the deployment and configuration of the test environment (e.g., hardware specification, software environment, network setting

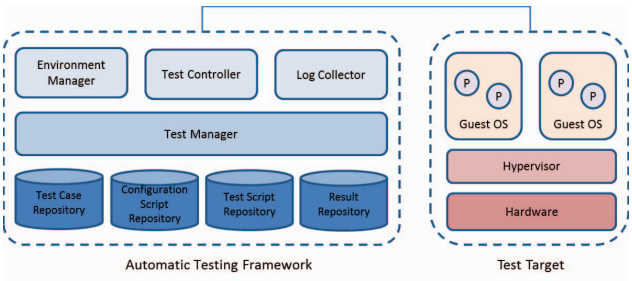

Fig. 1. Architecture of the automatic testing framework

978-1-4799-7377-4/14 $31.00 © 2014 IEEE
DOI 10.1109/ISSREW.2014.28

IEEE computer society

and so on), Test Controller is responsible for performing testing tasks. Test results and related logs are collected by Log Collector and updated to associated test cases.

Currently, the prototype of automatic testing framework is built by several open source software, including Jenkins, Testopia and JMeter. Jenkins is used to manage the life cycle of testing tasks, and to integrate with Testopia (test case management system) to retrieve related information and update test results. In addition, JMeter is used to simulate usage scenarios to the virtualization environment. Corresponding scripts are managed and executed by Jenkins to achieve testing tasks.

III. Major Features of The Framework

Based on the architecture of the framework, major features are designed and implemented to facilitate the functional and performance evaluation on virtualization environments.

A. Deployment and configuration of the test environment

The feature includes the deployment and configuration of the hardware specification, software environment and network setting of virtual environments. For instance, according to test case, virtual machines can be defined (e.g., memory size, disk image and network setting) and launched. In addition, corresponding software environment (e.g., web server, database, storage server and so on) can be configured for further testing tasks. Related settings can be described by XML, and Environment Manager will parse the document to perform corresponding deployment and configuration by scripts.

B. Deployment of usage scenario

Usage scenario is deployed to simulate actual usage of the virtualization environment. For instance, JMeter (open source web application testing tool) can be used in the framework to generate certain load (e.g., concurrent access to web server) on virtual machines. Different load and load types helps to reveal the reliability, availability and performance characteristics of virtualization environment. Similarly, usage scenario can be described by XML and parsed by Test Manager to perform the deployment tasks [5].

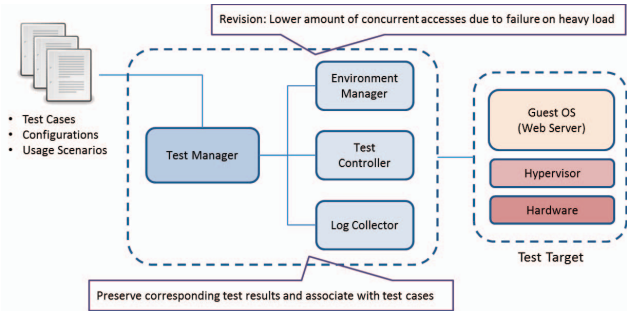

Fig. 2. Result preservation and test plan revision

C. Test result preservation and test plan revision

After the completion of test execution, the test result can be preserved to present the quality of virtualization environment efficiently. In the framework, Test Manager retrieves and performs particular test case, and associates corresponding test result (e.g., pass or fail) with that test case. In addition, summary and statistics of test run can be generated. Thus, developers and testers can efficiently get the insight of the quality and decide further development or maintenance tasks.

In addition, test plan revision is taken into consideration of the framework feature in order to decrease the redundant configuration effort and discover potential defects efficiently. As shown in Fig 2, if the test result of particular test case is fail, corresponding adjustment can be performed (e.g., lower the load on virtual machine, allocate more resource for virtual machine and so on). On the contrary, if the test result is pass, the test environment or usage scenario can be revised to more critical setting. The strategy of test plan revision can be described by XML to cover related adjustments. It helps to reveal the quality of virtualization environment automatically and efficiently.

IV. Conclusion

In order to facilitate the functional and performance evaluation on virtualization environments, an automatic testing framework is introduced in this paper. The framework provides automatic methods of test environment setting, usage scenario deployment and test plan revision to deal with complex and time consuming testing tasks. In addition, the framework is built by several open source software, which decreases the deployment cost and retains the flexibility. Further testing tasks will be performed (e.g., live migration and fault tolerance on virtualization environment) through the automatic testing framework to evaluate the usability and benefits.

Acknowledgment

This study is conducted under the Industrial Fundamental Technology Research Program in Electronic, Electrical and Software Fields (2/4) of the Institute for Information Industry which is subsidized by the Ministry of Economy Affairs of the Republic of China.

References

[1] K. Ye, J. Che, X. Jiang, J. Chen, and X. Li, "vTestkit: A Performance Benchmarking Framework for Virtualization Environments," *Proceedings of 2010 Fifth Annual ChinaGrid Conference*, pp. 130–136, July 2010.

[2] R. McDougall and J. Anderson, "Virtualization Performance: Perspectives and Challenges Ahead," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 4, pp. 40–56, Dec. 2010.

[3] W. Hu, A. Hicks, L. Zhang, E. M. Dow, V. Soni, H. Jiang, R. Bull, and J. N. Matthews, "vTestkit: A Performance Benchmarking Framework for Virtualization Environments," *Proceedings of the 2013 ACM Cloud and Autonomic Computing Conference*, Aug. 2013.

[4] K. Ye, Z. Wu, B. B. Zhou, X. Jiang, C. Wang, and A. Y. Zomaya, "Virt-B: Towards Performance Benchmarking of Virtual Machine Systems," *IEEE Internet Computing*, vol. 18, no. 3, pp. 64–72, May-June 2014.

[5] C. H. Kao, C. C. Lin, and J.-N. Chen, "Performance Testing Framework for REST-Based Web Applications," *Proceedings of the 2013 13th International Conference on Quality Software*, pp. 349–354, July 2013.