Alabdulhafez A, Ezhilchelvan P.

Experimenting on virtual machines co-residency in the cloud: A Comparative Study of Available Test Beds.

***In: 29th Annual ACM Symposium on Applied Computing*.**

2014, Gyeongju, Korea: ACM.

Experimenting on Virtual Machines Co-residency in the Cloud: A Comparative Study of Available Test Beds

Abdulaziz Alabdulhafez
Newcastle University
School of Computing Science, UK
Abdulaziz.alabdulhafez@ncl.ac.uk

Paul Ezhilchelvan
Newcastle University
School of Computing Science, UK
paul.ezhilchelvan@newcastle.ac.uk

ABSTRACT

Cloud computing aims to provide users with instant, on-demand access to large pools of computational resources. Although many organizations and end-users have migrated services and data to the cloud, there are a number of security concerns that exist there. In particular, recent researches have highlighted virtual machine (VM) co-residency as a new risk brought to the infrastructure as a service (IaaS) cloud by virtualization technologies which form the core of IaaS platforms. The placement of an attacker's virtual machine (VM) on the same physical server as a victim's VM is the key to successfully launching several harmful side-channel attacks to gain sensitive and valuable information about the co-residing VMs. When such placement manifests in a malicious manner then it is called VM co-residency. Studying these new VM co-residency's associated threats in the large, non-transparent and diverse IaaS clouds can become a very challenging task that requires an effective test bed that supports experimentation under different scenarios and settings. In this paper a number of cloud platforms and software tools are evaluated on their suitability as test beds for experimenting on VMs co-residency. It concludes with a recommendation for implementing a new VM co-residency simulator which can be used as a test bed for future research on VM co-residency in the cloud.

Categories and Subject Descriptors

D.2.8 [**Metrics**]: Performance Measures; I.6.4 [**SIMULATION AND MODELLING**]: Model Validation and Analysis.

General Terms

Reliability, Experimentation, Security

Keywords

Cloud computing, security, multi-tenancy, VMs co-residency, virtualization

1. INTRODUCTION

Cloud computing services, such as Microsoft's Azure [6] and Amazon's EC2 [1], provide infrastructure as a service (IaaS) allowing users to create and run their own servers in the cloud as virtual machines (VMs) on a pay-as-you-consume basis. With the emergence of cloud computing as a widely used solution for IT operations outsourcing, this hosting model inherits some significant and critical security issues. Among these issues is the risk of launching side-channel attacks on co-residing VMs by malicious users. They do this by exploiting the virtualization technology that is used for resources sharing between users in the IaaS cloud [29]. Cloud providers usually allow many tenants to run their own VMs on a shared physical infrastructure, known as "multi-tenancy". A side-channel attack requires placing a VM on the same physical server as the victim's VM, in order to successfully launch a side-channel attack, such as: a Theft-of-Service attack [22]; extracting a decryption key [46]; and many others [3] [4] [5] [9] and [15]. There is no single or best approach for studying VM co-residency in the cloud because there exists too many factors that need to be taken into account when conducting the experiments. These include cloud architecture, functional and non-functional requirements, etc. In addition, studying VM co-residency is of an experimental nature and therefore it requires a proper test bed that satisfies the design of experiment's test bed conditions [41]. Therefore, this paper makes the following contributions: First, exploring the available test beds which can be used to examine different aspects of VM co-residency, in order to help the researchers to choose the best option that fulfils their experiments' requirements. Second, comparing different test beds based on how they meet certain requirements for experimenting on large-scale clouds, such as scalability, cost, etc.

In this paper, section 2 outlines the test bed selection criteria to help identify the most suitable test bed for different types of VM co-residency experiments. In section 3, a survey of 20 of the available test beds for a VM co-residency experiment is provided, followed by a comparison and evaluation of the elected test beds according to the selection criteria. This leads to a conclusion of how suitable each test bed is for different type of experiments on VM co-residency in large IaaS clouds.

2. TEST BED SELECTION CRITERIA

In this paper, it is assumed that studying VM co-residency in large IaaS clouds requires experimentation with various cloud user types, settings and volumes, various kinds and numbers of VMs, different types and numbers of heterogeneous hosts and clusters, and most importantly a number of VM allocation algorithms. Performing such an experiment in large and dynamic environments needs a test bed that meets a certain set of criteria and requirements in order to conduct the experiment efficiently within a short time and under limited resources. For the purpose of conducting this evaluation, it has been assumed that the available time for experimentation on the selected test bed is 3 calendar months, and the research's available resources are a small lab which consists of 10s of mid-range machines operated by a single researcher. By applying the design of experiments theory principles [14], the criteria are set to help choosing the

most suitable test bed for this experiment. The first three criteria were inspired by [24], whilst the remaining were derived from [10], and directly relevant to the needs of the type of experiments under discussion. After reviewing each of the test beds, the final evaluation will examine each test bed against each of the selection criteria to be described in this section.

2.1 Repeatable and Controllable

Gaining a full control over a repeatable test environment is essential for sound, effective and accurate research and experimenting in the cloud. A repeatable experiment means that re-conducting the same experiment by the same experimenter must produce similar results. Needless to say, being able to conduct and repeat VM co-residency experiments in unpredictable environment conditions with full control of the cloud resources (e.g. cloud users, hosts, clusters, etc.) is the most important key to achieve meaningful results.

2.2 Reproducible

The test bed must produce the same results when the same experiment is conducted by a different researcher/operator.

2.3 Flexible

A flexible test bed must offer the ability to run the same experiment on several cloud platform architectures with different levels of details and different VM placement algorithms. This is crucial to allow the experiment results to be generalizable.

2.4 Easy to Use

The test bed is available and legal to use in experimental activities. Also, the time required for downloading and deploying the test bed with proper technical documentation defines the easiness requirement.

2.5 Scalable

VM co-residency experiments are usually designed to be conducted on various scales of cloud computing architectures. Scalability means that the chosen test bed is able to accommodate the increase in the size of cloud resources without losing performance, while maintaining the minimum expenditure of the research's resources.

2.6 Inexpensive and Not Time Consuming

In general, experimentation on large scale cloud computing architecture requires both time and computational resources. It is important when selecting the experiment's test bed, to consider the time and budget limitations for running the experiment on the selected test bed. Quick implementation of the experiment on the test bed, with minimum expense, as well as an acceptable execution speed are important factors that influence the test bed selection decisions.

2.7 Ability to Apply the Experiment's Input Values to the Test Bed

Since the experiment's design usually requires exploring the domain space by repeating the experiment under different input values, it is necessary to have an extensible test bed which allows easy control of the input values.

2.8 Sufficient Reporting/Monitoring System

Large-scale experiments usually produce a vast amount of output and statistical data that are used to analyse the results. In addition to the need for good reporting capabilities, the test bed must also allow the user to effectively monitor and record all necessary actions related to VM co-residency, such as the ability to detect VM co-residency easily and other co-residency related behaviours.

3. AVAILABLE TEST BEDS

The experimental validation methodologies in large-scale systems presented in [24] aims to define the best practices to conduct good experiments in large-scale systems. The suggested experimental methodologies are categorized based on the type of test bed they use. They include: real-platform experiments (executing real applications on real platforms); benchmarking (executing modelled applications on real platforms); emulation (executing real applications on modelled platforms); and simulation (executing modelled applications on modelled platforms). Looking at the above experimental methodologies, the real-platform and the benchmarking experiments usually use real platforms as a test bed, whereas the emulation and simulation methodologies use modelled platforms. Focusing on real-platform and simulation methodologies in this paper, three different test beds are selected for comparison based on the aforementioned test bed criteria: (1) real public IaaS platforms; (2) real private IaaS platforms; and (3) cloud computing simulators. The comparison between these three test beds is conducted as a straightforward evaluation to assess each test bed against each criterion. This comparison will help researchers to select the most suitable test bed for their experiments on VM co-residency in IaaS clouds.

3.1 Public IaaS Platforms

Cloud computing providers offer users the ability to rent computing infrastructure (e.g. servers) on-demand to cover their needs. Public cloud computing platforms such as Microsoft's Windows Azure, Amazon's EC2 and Rackspace [7] provide IaaS, allowing users to run their own servers in the cloud by simply creating VMs as servers, according to the providers' service level agreement (SLA). In order to utilize their physical infrastructure, virtualization is used to allow physical resources to be shared between users, and because of this, each cloud platform exhibits different workloads and can vary in the underlying infrastructure and configurations and compliance to certain SLAs.

Using public clouds as test bed is possible, yet it shows some limitations. Ristenpart et al pioneered research in "Hey, You, Get Off of My Cloud: Exploring Information Leakage in Third-Party Compute Cloud" uses Amazon's EC2 as a test bed, showing that it is possible to map the internal cloud infrastructure in order to locate where a specific targeted VM is likely to reside. After achieving this, new VM probes are launched until one or more of these probes become co-resident with the targeted VMs. They also describe a number of attacking scenarios where a malicious user can gather sensitive information from co-resident VMs that share the same underlying machine using cross-VM side-channel attacks [39]. Other researches, such as the AmazonIA paper [44], have used public cloud platforms as test beds. For instance, the researchers in AmazonIA have used Amazon's EC2 to launch various crafted Amazon Image Attacks in which they were able to collect very sensitive information (including credentials, passwords and keys). In addition, Amazon's EC2 also has been used as a test bed in [10]. Although the available public IaaS clouds, including Amazon's EC2, are usually easy to use with their rapid scalability (as well as the numerous amounts of

available documentations and how-to-use resources), using public clouds as a test bed comes with its own expenses. The diverse varieties of possible cloud infrastructure configurations and settings make the use of real cloud platforms as a test bed on VM co-residency a very expensive and time consuming task. This is because of the pay-as-you-go nature of the public cloud and the need for conducting repeatable tests with different configurations and settings, which results in a high number of experiment runs [17]. Furthermore, public cloud providers such as Amazon EC2 and Windows Azure usually obscure the details of their cloud infrastructure, networks and even VM placement policies, which results in a lack of transparency [39]. With little to no transparency, it becomes difficult to conduct testing experiments on such platforms because the testers cannot obtain the necessary information about the cloud anatomy and the implemented cloud policies, making the public cloud a non-repeatable and hard to control test environment. This also might result in the inability to implement a sufficient reporting system for detecting underlying events related to VM co-residency, as well as the inability to generalize the experiment's results due to the use of a very specific cloud architecture. Furthermore, the level of control given by the cloud provider to the user is usually very limited, which makes it difficult to set the experiment's input values. In some situations, it is also possible that this type of extensive experimental usage might lead to a violation of the cloud's usage policy [1]. From what has been discussed before, this combination of limitations shows that public cloud computing platforms are thought not to be always the best test bed for this type of research.

3.2 Private IaaS Platforms

Private IaaS cloud platforms, such as the open-source Eucalyptus private cloud platform [36] and OpenNebula [19] offer similar functionalities as public IaaS platforms, except for one major difference: private IaaS platforms can be implemented in the user's own physical infrastructure whereas public IaaS run on a third party infrastructure. This feature of the private IaaS offers more flexibility to implement and model a vast array of possible cloud architectures. Moreover, an open-source private cloud gives the researchers the power to control and monitor every single event in their experiments, which forms a good test bed to both fully control the experiment's input variables and to allow the tester to implement a decent reporting system to monitor all VM co-residency related events in a repeatable and controllable test environment. Also, private cloud platforms have been used as test beds in experimental research context for various objectives. For instance, [28] have conducted an evaluation of software ageing effects on the Eucalyptus cloud computing infrastructure, whilst other researchers have used Eucalyptus as a proof of concept of autonomic resource provisioning in rocks clusters [21]. However, there is still a need when using private cloud platforms for large capital investment to purchase and maintain the required hardware infrastructure to conduct scalable experiments, which can sometimes exceed the available resources for researchers.

3.3 Discrete-Event Simulators

One of the widely used test beds in large-scale experiments is to use discrete-event simulators, such as grid simulators and cloud computing simulators, instead of using real cloud platforms as test beds. A discrete-event simulation [13], as opposed to real-time simulations that mimic physical systems' execution at the exact rate as actual clock time, has a collection of state variables which reflect the current system status. These state variables can change only at discrete instants (called events), whose sequential order describes the simulated system behaviour. A list of some of these simulators with descriptions and comparisons is provided next.

3.3.1 Grid Simulators.

In the area of distributed computing, grid computing is described as a set of distributed systems that can provide on-demand access to dependable, consistent and inexpensive hardware and software infrastructure to process large amounts of non-interactive workloads [23]. There are many multi-tier data centre simulation platforms, such as MDCSim [42], that have been designed to support the modelling of different hardware specifications of the common data centres' components, including servers, network switches and communication links. However, grid simulators require more advanced capabilities in order to simulate the distributed applications' behaviour more accurately. In order to meet the demand of research and development on grid systems, several grid simulators, such as SimGrid [18], MicroGrid [43], GridSim [16] and GangSim [20], have been introduced. In late 2012, SimGrid started to support a very basic interface to implement virtualization environments, however this interface is highly experimental as stated on the project website and that they "…do not expect too much of it right now" [8]. Among these grid simulators, GridSim is the most related to VM co-residency research as it has been extended to form the base of some of the cloud simulators [17].

Initially, GridSim was introduced as a simulator for resource modelling, application scheduling and performance analysis in grid computing environments. It supports the modelling of various application models and it is capable of automating the task of generating a stream of application workloads. GridSim was built upon SimJava [31], a process-based discrete-event simulation framework implemented in Java. Since SimJava runs a unique thread for each element in the simulation, it has been shown in [37] that SimJava toolkit performance degrades when simulating more than 2,000 grid entities concurrently, because of the high consumption of memory.. Since GridSim implements in the exact way in which SimJava simulates the grids, it inherits this scalability limitation. Even though grid simulators have been designed to effectively and comprehensively model grid environments and systems to the maximum extent, none of them are capable of clearly abstracting the application layer from the virtual and physical machines layer, which is the core requirement of any cloud computing environment. This type of abstraction is required when trying to model multi-layer architecture such as the IaaS cloud. In addition, the above grid simulators lack the capability to model virtualized resources and applications, as well as the cloud management environment [40]. Therefore, it is not practical to use grid simulators in this type of experiment and to use cloud computing simulators instead.

3.3.2 Cloud Computing Simulators.

A cloud computing simulator is a toolkit that models and simulates different cloud computing elements and environments [33]. Cloud simulators are usually capable of simulating multiple data centres, modelling the creation of VMs and the allocation of these VMs to hosting machines, as well as the creation of cloud users and generating different types of cloud-related requests and many other elements of cloud computing. The use of cloud simulators can provide a higher degree of flexibility to conduct different types of experiments on a close-to-real cloud environment. Several cloud computing simulators are reviewed

next in order to include them in the evaluation at the end of this paper.

3.3.2.1 CloudSim.

CloudSim [17] is one of the widely used cloud computing modelling and simulation toolkit which was developed at the University of Melbourne, Australia. The main goal of CloudSim is to help cloud computing researchers to conduct comprehensive simulation-based experiments. The main features that CloudSim offers includes the modelling and simulation of large scale cloud computing IaaS, with configurable data centres, physical nodes, resources and virtualization provisioning, as well as power management. With its multi-layer design framework that reflects the layered architecture of real cloud computing environments, CloudSim was developed using Java and was built on top of the SimJava-based grid simulator GridSim. As disabused earlier, GridSim has several scalability limitations which CloudSim inherited initially. Therefore, the developers of CloudSim decided to modify the first release of this simulator and implement a new discrete-event management framework. This became the CloudSim core simulation engine (Figure 1). The new framework uses only three main threaded components, and the remaining entities are implemented as objects. Each component in the CloudSim architecture is implemented as a Java class that can be extended or changed to reflect certain simulation requirements.

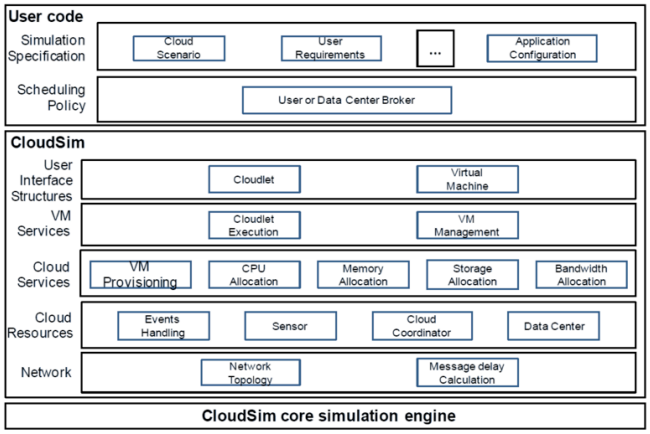

Figure 1. CloudSim Architecture

Difficulties arise, however, when an attempt is made to simulate certain cloud environments with specific requirements using CloudSim. Each of these different difficulties forms a reason behind the development of many successive simulators that have been built upon CloudSim. At least four cloud computing simulators worldwide have been adopted to extend CloudSim in order to add new functionality or components that CloudSim is missing, such as network latency, bandwidth simulation, SLA management, and more. For example, [27] highlights the need to adopt an easy-to-set-up and user-friendly cloud platform, in order to be used in education environments. They have surveyed the available cloud simulators in the market and elected CloudSim as a base platform for their intended simulator. They claim that new enhancements and extensions to CloudSim are essential to maintain a decent cloud computing educational toolkit. These extensions have been implemented in the TeachCloud cloud computing simulator. TeachCloud features a new graphical user interface (GUI) for CloudSim, adding SLA management and business process management modules on the architecture level, as well as building several cloud network models such as VL2, BCube, Portland and DCell to model different topologies that can be found in real cloud environments.

Moreover, a group of researchers at the Pontifical Catholic University of Rio Grande do Sul in Brazil have recently introduced another cloud simulator and visual modeller based on CloudSim, called CloudAnalyst [45]. The primary goals of CloudAnalyst are to visually model, simulate and analyse the effects of geographic distribution of large distributed social network applications under multiple deployment configurations in the cloud, in order to give large applications' developers helpful insights into how to effectively distribute these type of applications. Using CloudSim as the base simulation engine, CloudAnalyst leverages whole features of CloudSim, and implements important functionality that is missing.

For example, instead of spending unnecessary time on programming the simulation environment requirements using CloudSim, CloudAnalyst provides the user with a GUI to easily control the simulator variables This helps the user to focus on the environment simulation experiment. The rest of the added functionality is mainly intended to introduce a basic network, bandwidth and latency modelling management. Thus allowing the user to configure the amount of generated applications' workloads, to supply some information of the geographic distribution of the origin of the generating traffic, as well as defining the data centres' locations. By using this detailed information, CloudAnalyst is capable of simulating distributed applications' behaviour in the cloud, as well as producing various graphical reports in the form of tables and charts of users' requests response time, requests processing time and other useful analytical data.

CloudReport [2] is another CloudSim-based cloud computing simulator developed at Federal University of Ceara, Brazil. Its functionalities are very similar to CloudAnalyst, providing an easy-to-use GUI and a rich reporting module.

Similar to CloudAnalyst, yet with more architecture-level changes, NetworkCloudSim cloud computing simulator [40] has been introduced to overcome the limitations that can be found in CloudSim's network layer. CloudSim's network layer views the data centre's resources as a collection of VMs, and therefore it is capable of simulating limited communications activities between resources. The developers of NetworkCloudSim argue that CloudSim suffers when simulating a large distributed application (such as message passing parallel applications or multi-tier web applications hosted in different machines), where a precise evaluation of resource allocation algorithms require a more sophisticated modelling of the data centre's interconnection network. They also claim that they have equipped NetworkCloudSim (figure 2) with the most advanced realistic application model compared to CloudSim, where they "... have designed a network flow model for Cloud data centres utilizing bandwidth sharing and latencies to enable scalable and fast simulations."

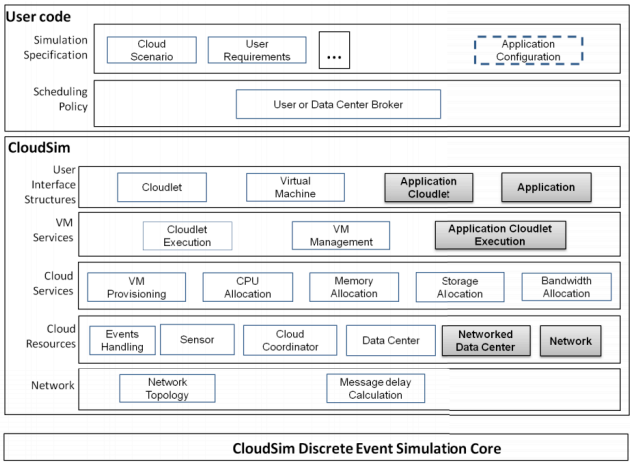

Figure 2. NetworkCloudSim's new elements introduced to CloudSim Architecture

3.3.2.2 GreenCloud.

In recent years, there has been an increasing amount of literature on energy-aware cloud computing date centres. Researchers in this area have started to adopt the use of cloud computing simulators to experiment with different environment-friendly resource allocation algorithms, to utilise the computing resources in an energy-efficient fashion [32]. As an extension of the well-known NS2 network simulator [25], GreenCloud was first introduced in 2010 as a packet-level simulator for energy-aware cloud computing data centres [30]. Together with the workload generation and distribution which GreenCloud offers, the simulator's primary task is to precisely capture the energy consumption readings of the data centre components (hosts, switches and links) as well as any communication patterns in the packet-level. Moreover, it can simulate and produce the simulation results for two-tier and three-tier architectures. GreenCloud's core strength can be observed in its ability to model the communication interactions of any data centre network with an extensive level of detail, since it uses the NS2 to implement a full TCP\IP protocol model. However, this advantage can affect GreenCloud's by limiting its scalability due to the heavy memory requirement needed to simulate such detailed models.

3.3.2.3 GroudSim .

Similar to CloudSim, GroudSim is a Java-based discrete-event cloud computing simulator developed by [38]. In contrast to CloudSim and the aforementioned cloud computing simulators, GroudSim is capable of supporting the simulation of applications running on combined cloud and grid platforms. Its developers claim that it offers better scalability and performance compared to related process-based simulators, since it uses discrete-event simulation. GroudSim presents some basic analysis and statistics of the simulated system. It also supports the modelling of grid and cloud infrastructures including network and computational resources, task scheduling, file transfer, and cost, failure and background models. Nevertheless, GroudSim has not escaped criticism from its developers, as they state in [37] that although it has been successfully used in a previous scientific work, the drawback, was that further programming needs to be done in order to implement the simulation experiment by using a different interface from the one used in the real application, which extends the required efforts to execute the experiments.

3.3.2.4 Koala

As a medium-scale discrete-event simulation of IaaS, Koala is a project run by the National Institute of Standards and Technology (NIST) with the aim to implement a cloud computing simulator that serves the research on cloud in a more controllable environment [34]. High accuracy models require the definition of many parameters and lead to long run-times resulting in more realistic simulation results, whereas the opposite is true for high abstraction models. Koala has been designed to simulate cloud environments with some abstractions while maintaining a good level of model accuracy. Offering a multi-layered architecture (figure 3) based on the commercial discrete-event simulation environment SLX [26], Koala was designed to model the Amazon EC2's architecture through the use of Eucalyptus APIs.

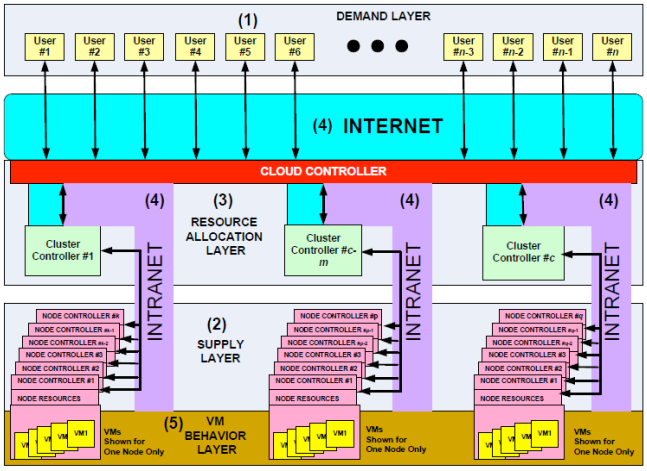

Figure 3. Koala architecture

Koala is capable of simulating several essential cloud computing components, such as cloud controller, cluster controller and node controller, where they all communicate using web services. Initial sensitivity analyses using Koala as a test bed [33] identified the number of cloud users, number of clusters and number of nodes per cluster as the major parameters that influence the simulator behaviour. Perhaps the most interesting feature of Koala (which has a relation to VM co-residency's experiments) is that it has several resource allocation algorithms implemented in the cloud controller, including least-full first, next-fit, first-fit, most-full first, percent allocated, random and tag-and-pack. Unfortunately, NIST's project would have been more useful for this type of research if Koala's developers had made this simulator available for the researchers to use. This forms the key issue that might be a strong obstacle that prevents researchers from considering Koala as a suitable test bed for VM co-residency experiments.

3.3.2.5 iCanCloud.

Very much like the Koala simulator, the iCanCloud simulation toolkit was specifically implemented to simulate cloud resources as if they are really running in the Amazon Elastic Compute Cloud (EC2). It can also be extended to simulate other cloud platforms, as its implementers claim with a primary aim "… to predict the trade-offs between cost and performance of a given application executed in a specific hardware, and then provide

users with useful information about such costs." [11]. Originally built upon the distributed systems simulator, SIMCAN [12], iCanCloud adopts a multi-layer system design that models the common cloud computing stack.

With its user-friendly GUI and the ability to generate graphical reports, iCanCloud simulator easily allows the addition of new cloud components into its repository. Unlike GroudSim simulator, iCanCloud provides a POSIX-based API for modelling the simulation applications in a much easier way. In addition to the fact that Amazon's E2C is the only environment which is modelled in iCanCloud, perhaps the most serious disadvantage of this simulator is that it does not provide a module to take care of creating the cloud resources, such as users, hosts, and VMs, at the start of each simulation run. Instead, it requires use of the provided GUI to manually define the new cloud resources parameters one by one, which appears to be impractical when modelling a large scale cloud environment.

4. EVALUATION AND DISCUSSION

The first and foremost decision need to be made when experimenting with VM co-residency in the cloud is to select the appropriate test bed that meets the experiment's requirements and constraints. Whether you select a physical test bed or/and a simulator, each option is most suitable in different scenarios and different situations. In this paper, 20 of available test beds for experimenting on VM co-residency in the cloud have been discussed, including physical test beds (i.e. public IaaS and private IaaS platforms) and simulators. The evaluation matrix of these test beds is presented in table 1.

Simulators are usually capable of modelling several essential cloud computing components with some abstractions, while maintaining a good level of model accuracy. Simulators can be a sensible option when experimenting on very large-scale and dynamic clouds when there is a need to be able to control and monitor the simulated cloud's behaviour. While the aforementioned cloud computing simulators vary in satisfying the test bed criteria defined earlier, one major criticism is that none of the discussed simulators implements sufficient VM co-residency monitoring, detection and reporting modules, which are critical when studying VM co-residency in the cloud. Implementing these modules into these existing simulators is not an option for closed source simulators. On the other hand, introducing these modules to the open source simulators is possible, but requires a considerable amount of time and efforts to achieve; especially when each of the discussed simulators focuses on modelling cloud elements unrelated to this type of experiment. In addition, some simulators are platform independent (e.g. Java-based simulators) but relatively slow in execution.

After exploring the available physical test beds (i.e. public and private IaaS platforms) for VM co-residency experiments, the results obtained from experiments that have been carried out using physical test beds are usually more accurate than when using simulators, as they are "real" platforms. However, both public and private IaaS platforms have shown to suffer from a number of shortcomings. For instance, public IaaS platforms are often not reproducible test beds, whereas extensibility and repeatability are hard to achieve. Private IaaS platforms in particular can be an expensive option when the experiment needs to be conducted on a large and scalable cloud environment. It is worth mentioning that [24] confirms that "experiments on real platforms are often not reproducible, whereas, extensibility, applicability and revisability are hard to achieve".

Alternatively, satisfying all test bed criteria can be achieved by designing and implementing a new flexible discrete-event cloud computing simulator that solely focuses on modelling all the behaviours of VM co-residency and its related modules in a way that supports the run of this type of research experiment in a fully controllable and repeatable environment. In fact, implementing and using a custom simulator instead of relying on an existing simulation tool has become a sensible practice for satisfying each individual research's requirements. [35] analysed 141 papers that use simulation to study large-scale peer-to-peer systems and reported that 30% of these papers use their own custom simulation tool.

Criteria	Real Platforms		Simulators	
	Public IaaS	Private IaaS	Grid	Cloud
Controllable	No control on infrastructure and resource allocation	Full Control		
Repeatable / Reproducible	Unknown settings and location in which the experiment is run in the cloud	Yes		
Flexible	Very limited	Yes	In open source simulators only	
Easy	Yes with friendly web-based GUI and instant support	Requires self-maintained infrastructure	Yes when support, documentation and GUI are provided	
Scalable	Yes, add as many resources as needed	Limited hardware infrastructure (e.g. expensive to increase the number of resources)	Limited in Java-based simulators that use threading	
Cost and Time	Pay per use, on demand	Requires investing on physical infrastructure, takes time for deployment and maintenance	Possible to run immediately on a single machine – cost is limited to the license fee (if required) - virtual resources can be added instantly with no cost	
Controllable inputs	Not possible	Yes to some extent (e.g. changing inputs may require extra resources)	Yes - virtual resources can be changed instantly with no cost	
Reporting/ Monitoring	Not possible for VM co-residency	Requires implementation		

Table 1. Comparison of test beds and how they satisfy the defined criteria for evaluating a given test bed suitability for conducting VM co-residency experiments

5. CONCLUSION

In this paper, 20 different cloud platforms and software tools have been examined on their suitability as a test bed for VMs co-residency. These test beds have been categorized into public IaaS platforms, private IaaS platforms and discrete-event simulators. These test beds have been selected based on their popularity, availability of documentation and support, and whether they are applicable for cloud experimental usage. The selected test beds have been evaluated against seven criteria such as their capabilities and flexibilities in modelling an IaaS cloud, and for input control as well as output analysis. Using simulators can be useful and more effective, especially if physical test beds (public and private IaaS platforms) are expensive or not feasible. However, the evaluation shows that none of the current simulators can be easily utilized for VM co-residency related research. Therefore, the future work will consist of extending the comparison to include benchmarking and emulation test beds, as well as designing and implementing a discrete-event VM co-residency simulator that allows the modelling of cloud computing environments and also can simulate and monitor the VMs behaviour in more depth. It is hoped this VM co-residency simulator will form a suitable test bed that helps in advancing researches on this very important topic.

6. ACKNOWLEDGMENTS

Our thanks to King Saud University, Saudi Arabia for their generous support to this research.

7. REFERENCES

[1] Amazon EC2, 2013.

[2] CloudReport, 2012.

[3] CVE-2007-4993. pygrub (tools/pygrub/src/grubconf.py) in xen 3.0.3.

[4] CVE-2007-5497. Multiple integer overflows in libext2fs.

[5] CVE-2010-2240. The do_anonymous_page function in mm/memory.c.

[6] Microsft's Windows Azure, 2013.

[7] Rackspace Open Cloud, 2013.

[8] SimGrid Project Website., 2013.

[9] VMSA-2008-0008. Updates to VMware Workstation, VMware Player, VMware ACE, VMware Fusion Resolve Critical Security Issues., 2008.

[10] Alabdulhafez, A. and Ezhilchelvan, P. Analyzing the Success Rate of Virtual Machines Co-residency in the Cloud. in *The 6th Saudi Scientific International Conference Saudi Scientific International Conference (SIC)*, (Brunel, United Kingdom, 2012), 164-168.

[11] Alberto, N., ez, Jose, L.V., zquez, P., Agustin, C.C., Gabriel, G.C., Jesus, C. and Ignacio, M.L. iCanCloud: A Flexible and Scalable Cloud Infrastructure Simulator. *J. Grid Comput.*, *10* (1). 185-209.

[12] Alberto, N., Javier, F., Rosa, F., Félix García, C. and Jesús, C. SIMCAN: A flexible, scalable and expandable simulation platform for modelling and simulating distributed architectures and applications. *Simulation Modelling Practice and Theory*, *20*.

[13] Banks, J. *Discrete-event system simulation*. Prentice Hall, 2001.

[14] Box, G.E.P., Hunter, J.S. and Hunter, W.G. *Statistics for experimenters: design, innovation, and discovery*. Wiley-Interscience, 2005.

[15] Brodkin, J. VMware confirms source code leak, LulzSec-affiliated hacker claims credit, 2012.

[16] Buyya, R. and Murshed, M. GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing. *Concurrency and Computation-Practice & Experience*, *14* (13-15). 1175-1220.

[17] Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A.F. and Buyya, R. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software-Practice & Experience*, *41* (1). 23-50.

[18] Casanova, H. Simgrid: a toolkit for the simulation of application scheduling. *First Ieee/Acm International Symposium on Cluster Computing and the Grid, Proceedings*. 430-437.

[19] Dejan, M. OpenNebula: A Cloud Management Tool. Ignacio, M.L. and Ruben, S.M. eds., 2011, 11-14.

[20] Dumitrescu, C.L. and Foster, I. GangSim: A simulator for grid scheduling studies. *2005 IEEE International Symposium on Cluster Computing and the Grid, Vols 1 and 2*. 1151-1158.

[21] Ekasit, K. and Sornthep, V. Autonomic resource provisioning in rocks clusters using Eucalyptus cloud computing *Proceedings of the International Conference on Management of Emergent Digital EcoSystems*, ACM, Bangkok, Thailand, 2010.

[22] Fangfei, Z., Goel, M., Desnoyers, P. and Sundaram, R., Scheduler Vulnerabilities and Coordinated Attacks in Cloud Computing. in *Network Computing and Applications (NCA), 2011 10th IEEE International Symposium on*, (2011), 123-130.

[23] Foster, I. and Kesselman, C. *The Grid: Blueprint for a New Computing Infrastructure*. Elsevier, 2004.

[24] Gustedt, J., Jeannot, E. and Quinson, M. Experimental Methodologies for Large-Scale Systems: a Survey. *Parallel Processing Letters*, *19* (3). 399-418.

[25] Issariyakul, T. and Hossain, E. *Introduction to Network Simulator NS2*. Springer, 2008.

[26] James, O.H. An introduction to SLX *Proceedings of the 28th conference on Winter simulation*, IEEE Computer Society, Coronado, California, USA, 1996.

[27] Jararweh, Y., Alshara, Z., Jarrah, M., Kharbutli, M. and Alsaleh, M.N. `TeachCloud: A Cloud Computing Educational Toolkit *The 1st International IBM Cloud Academy Conference*, North Carolina, USA, 2012.

[28] Jean, A., Rubens, M., Paulo, M., Rivalino, M. and Ibrahim, B. Experimental evaluation of software aging effects on the eucalyptus cloud computing infrastructure *Proceedings of the Middleware 2011 Industry Track Workshop*, ACM, Lisbon, Portugal, 2011.

[29] Keiko Hashizume, E.B.F., Nobukazu Yoshioka. Misuse Patterns for Cloud Computing *The 2nd Asian Conference on Pattern Languages of Programs*, Tokyo, Japan., 2011.

[30] Kliazovich, D., Bouvry, P., Audzevich, Y. and Khan, S.U. GreenCloud: A Packet-level Simulator of Energy-

aware Cloud Computing Data Centers. *2010 Ieee Global Telecommunications Conference Globecom 2010*.

[31] Kreutzer, W., Hopkins, J. and van Mierlo, M. SimJAVA - A framework for modeling queueing networks in Java. *Proceedings of the 1997 Winter Simulation Conference*. 483-488.

[32] Liu, J., Zhao, F., Liu, X. and He, W.B. Challenges Towards Elastic Power Management in Internet Data Centers. *Icdcs: 2009 International Conference on Distributed Computing Systems Workshops*. 65-72.

[33] Mills, K., Filliben, J. and Dabrowski, C. Comparing VM-Placement Algorithms for On-Demand Clouds *Proceedings of the 2011 IEEE Third International Conference on Cloud Computing Technology and Science*, IEEE Computer Society, 2011.

[34] Mills, K., Filliben, J. and Dabrowski, C., An Efficient Sensitivity Analysis Method for Large Cloud Simulations. in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, (2011), 724-731.

[35] Naicken , S.a.B., A. and Livingston , B. and Rodhetbhai , S. and Wakeman , I., Towards Yet Another Peer-to-Peer Simulator. in *Proceedings of The Fourth International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks*, (Ilkley, UK, 2006).

[36] Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L. and Zagorodnov, D., The Eucalyptus Open-Source Cloud-Computing System. in *Cluster Computing and the Grid, 2009. CCGRID '09. 9th IEEE/ACM International Symposium on*, (2009), 124-131.

[37] Ostermann, S., Plankensteiner, K., Bodner, D., Kraler, G. and Prodan, R. Integration of an Event-Based Simulation Framework into a Scientific Workflow Execution Environment for Grids and Clouds. *Towards a Service-Based Internet*, *6994*. 1-13.

[38] Ostermann, S., Plankensteiner, K., Prodan, R. and Fahringer, T. GroudSim: An Event-Based Simulation Framework for Computational Grids and Clouds. in Guarracino, M., Vivien, F., Träff, J., Cannatoro, M., Danelutto, M., Hast, A., Perla, F., Knüpfer, A., Martino, B. and Alexander, M. eds. *Euro-Par 2010 Parallel Processing Workshops*, Springer Berlin Heidelberg, 2011, 305-313.

[39] Ristenpart, T., Tromer, E., Shacham, H. and Savage, S. Hey, You, Get Off of My Cloud: Exploring Information Leakage in Third-Party Compute Clouds. *Ccs'09: Proceedings of the 16th Acm Conference on Computer and Communications Security*. 199-212.

[40] Saurabh Kumar, G. and Rajkumar, B. NetworkCloudSim: Modelling Parallel Applications in Cloud Simulations *Proceedings of the 2011 Fourth IEEE International Conference on Utility and Cloud Computing*, IEEE Computer Society, 2011.

[41] Schultz, M.K., Inn, K.G.W., Lin, Z.C., Burnett, W.C., Smith, G., Biegalski, S.R. and Filliben, J. Identification of radionuclide partitioning in soils and sediments: Determination of optimum conditions for the exchangeable fraction of the NIST standard sequential extraction protocol. *Applied Radiation and Isotopes*, *49* (9-11). 1289-1293.

[42] Seung-Hwan, L., Sharma, B., Gunwoo, N., Eun Kyoung, K. and Das, C.R., MDCSim: A multi-tier data center simulation, platform. in *Cluster Computing and Workshops, 2009. CLUSTER '09. IEEE International Conference on*, (2009), 1-9.

[43] Song, H.J., Liu, X., Jakobsen, D., Bhagwan, R., Zhang, X., Taura, K. and Chien, A. The MicroGrid: A scientific tool for modeling Computational Grids. *Sci. Program.*, *8* (3). 127-141.

[44] Sven, B., Stefan, N., rnberger, Thomas, P., ppelmann, Ahmad-Reza, S. and Thomas, S. AmazonIA: when elasticity snaps back *Proceedings of the 18th ACM conference on Computer and communications security*, ACM, Chicago, Illinois, USA, 2011.

[45] Wickremasinghe, B., Calheiros, R.N. and Buyya, R. CloudAnalyst: A CloudSim-based Visual Modeller for Analysing Cloud Computing Environments and Applications. *2010 24th Ieee International Conference on Advanced Information Networking and Applications (Aina)*. 446-452.

[46] Yinqian, Z., Ari, J., Michael, K.R. and Thomas, R. Cross-VM side channels and their use to extract private keys *Proceedings of the 2012 ACM conference on Computer and communications security*, ACM, Raleigh, North Carolina, USA, 2012.