Social Network Enhanced Cloud Resource Sharing

Zahra Ali[1,2] Raihan ur Rasool [3] Peter Bloodsworth[4] Shamyl Bin Mansoor[1]

National University of Science and Technology Islamabad, Pakistan[1]
University of Management and Technology Lahore, Sialkot Campus Pakistan [2]
Victoria University Melbourne, Australia [3]
University of Oxford, UK [4]
raihan.rasool@live.vu.edu.au, peter.bloodsworth@cs.ox.ac.uk

***Abstract*—Sharing cloud resources between groups of users is a challenge. Cloud providers do not commonly support users in sharing their spare dedicated resources with others. In developing countries, it is often too expensive for people to acquire a virtual machine of their own. Users may, therefore, wish to manage costs and increase computational resource usage by sharing their instances with others. This paper presents a container based Cloud Resource Bartering (CRB) model for sharing user's computational resources through a social network. In our approach, we have integrated a Facebook account with the computational cloud to enable tenants to share their unused cloud resources with other users. The performance of the proposed prototype is evaluated under different workloads. Based on our experimental results we conclude that the proposed model is well suited for the creation of a low-cost social cloud in developing countries.**

Keywords- Cloud resource bartering; Social cloud; Resource sharing in cloud; Enterprise virtual organization; Green social cloud;

1. Introduction

Developing countries face many social and economic challenges. Education is often seen as a key means of reducing the levels of poverty and re-enforcing economic growth in such countries. Unfortunately, due to a lack of funding, governmental and non-profit charitable institutes are not able to provide sufficient learning resources to students. Many institutes have limited access to basic research and development resources. This often significantly compromises the educational progression of students in developing countries. Emerging technologies tend to be expensive, at least initially and therefore may require considerable funding to be put in place before students can even begin to benefit from working with them. Cloud computing is one such technology that requires significant investment. It is increasingly becoming a vital part of the research infrastructure in many data-intensive fields. In developing countries specifically in most educational institutes, Cloud resources are not commonly made available to students and researchers because of the associated costs. If such countries are to develop and grow, then new ways of providing access to emerging technologies need to be found.

This paper focuses on how cloud resources can be made available to students in developing countries using an existing Facebook account. The major contribution of this paper is to present a container based cloud resource bartering (CRB) model for sharing cloud resources among users in emerging economies. The main aim is to break down several barriers that currently prevent students and small-scale communities from accessing the latest technology in such regions. To solve this problem, we have designed a cloud resource bartering (CRB) model that supports container based resource sharing among users. The proposed CRB model has been implemented and deployed on Facebook. Linux container (LXC) based sharing in EC2 may help the user to share a part of the complete resource with other tenant users. This is likely to also help the landlord users to manage the cost of renting their computational resources.

In this paper, we employ a range of different workloads to test the stability of container-based virtualization which enables the social community to share virtualized containers with each other. In addition to this, we have evaluated the implemented model and examined the stability and performance of shared cloud resources. For this purpose, we installed different workloads in LXC containers created in EC2 instance and observed its stability among a shared community. Our results show satisfactory performance of shared instance under different workloads. A community-based survey was carried out to gather qualitative feedback from real end users of the system. The result of this analysis confirms that social networks can play a vital role in cloud resource bartering and such approaches can work well in developing countries.

The contribution of the paper is twofold; firstly, the CRB creates a trusted community environment on Facebook, where users can share their idle compute resources with others. Secondly, CRB requires the sharer and the tenant to both view and accept the Social service level agreement (SSLA) before sharing. It also keeps a list of SSLA violation details and black list the users who have not followed the terms and conditions mentioned in SSLA. Both of these features have a novelty with respect to sharing resources on a social network according to the understanding of the author. To the best of our knowledge, it is the first effort in this direction.

The paper is organized as follows. Section 2 provides a comprehensive related review. Section 3 presents the methodology and implementation details of the CRB model. Section 4 demonstrates the effectiveness of the system through a detailed performance evaluation. Finally, section 5 concludes the discussion and provides some pointers to future work.

2. RELATED WORK

The importance of social networking has been observed in various scientific domains as a means of facilitating teamwork across different locations. Increasingly social networks are being used to synchronize research communities as well. One such example is MyExperiment [1] a platform for biologists, that creates a virtual research environment for sharing scientific workflows among collaborators. Similar scientific projects are Kepler [2] and Galaxy [3]. There are many projects that are integrating Grid computing and social networking technologies. For example, PolarGrid [4] shares polar ice sheet information and resources amongst members by using a web portal. This portal extracts data and shares information in multiple social networks to support global collaborations with the help of OpenSocial [5]. In the context of social networking, a number of techniques have been proposed for resource sharing. For example, the Automated Service Provisioning Environment (ASPEN) [6] is one such system that provides an integration of Web 2.0, social networking and cloud computing technologies. ASPEN exposes applications and shares data within an enterprise using a social network.

The models described previously focus on the specific research communities they serve and on the social aspects of collaboration between researchers. We can, therefore, suggest that existing social networks are potentially suitable platforms for sharing cloud resources in a more general way. There are many good examples of cloud and social networks working together, but in most cases, the cloud is simply hosting the applications of social networks, for example, a user can develop a Facebook application hosted by Amazon's AWS service.

There is some literature related to creating a Cloud infrastructure leveraging Social networking. A prototype is proposed in [7] for an economics based posted price model in a social cloud and an auction-based model in the computing cloud. The proposed model is for the storage of resources in the cloud, but this work is not implemented for any commercial cloud platform at present. The cloud offers far greater prospects in terms of shared processing power but there is a clear need for an application that can help users to share their resources with their friends by using a sharing model (e.g. resource bartering).

For scientific projects which require large-scale computing resources, one useful aspect of collaboration is the sharing of resources among project members. Social Cloud [8] represents a framework for Social Cloud computing with a focus on collaboration and resource sharing within a scientific community. The Social Cloud supports individuals or institutions by allowing them to contribute spare capacity by means of virtual machines leased through the social network. Members of the Social Cloud can contribute, request, and use virtual machines from other members. A Virtual Organization is formed among groups of members. Current work on the social cloud does not, however, support an established economic model for resource sharing. Such models might, for example, include an incentive mechanism that encourages a more diverse group of participants.

Previously, however, a range of different economic models have been created for Grid and Cloud resource sharing [9][10][11] as a means of encouraging interaction among producers and consumers.

Numerous computing platforms including distributed databases, clusters and grids are using market-based resource management systems. These systems adopt economic models such as an auction, bartering, commodity, and price based models. Economic models can be divided into two important categories based on whether the exchange methods use pricing[1]or bartering[2]. The industry has successfully adopted many pricing methods for resource sharing [12] but bartering models have not been popular because of lack of trust among rational users[3] (Cloud provider or Grid resource providers). In large distributed systems various computation efficient resource bartering techniques are proposed in the presence of rational users. The proposed CRB model, therefore, may help a community of friends to barter cloud resources without money necessarily changing hands.

Bartering models are classified into three different categories known as TiT for TaT (BitTorrent (2003) [13]),Volunteer Computing Seti@Home (2004) [14] and Network of Favors (OurGrid (2003) [15]). OurGrid is a network of favors that supports the community in site sharing. A site donates its idle resources as a favor based on a centralized priority. However, the proposed CRB model is providing this prioritization in a completely decentralized manner similar to systems like the Grid Architecture for Computational Economy (GRACE) [16] and the Compute Power Market [17]. In a similar way to BitTorrent, the CRB model may help to decrease the load on congested cloud providers by tracking the actions of users as a user is not anonymous in the proposed system. In our work, we have decided therefore to create a model that tracks the activities of users. It can be observed in the literature that many of the currently proposed models are not practically implemented whereas the proposed CRB Model is indented for use in a commercial cloud provider Amazon EC2.

There are many cloud-based applications that help service providers to support users and ensure that the cloud services offered are secure, reliable, and available. In [18] Provision, Assurance, and Auditing (PAA) provides such support in creating a highly dynamic cloud-based application that can respond to the new requirements without the need for recording and republishing the application. These kinds of models are considered to be adaptable for cloud-based social network applications and enable highly reliable and secure applications .

---

[1]The exchange value of resources is computed relative to a common form of currency
[2] Bartering allows users to exchange resources without money changing hands
[3] A rational user can be a buyer or seller of resources.

Cloud computing has fundamentally changed the concept of how services are built and delivered via a network. It has introduced a new paradigm known as cloud service engineering [19], that is providing a systematic approach for value-added services on the top of cloud computing infrastructure. In the past, many applications have been proposed that focus on supporting cloud service engineering. One of such applications is eternal cloud computation application development [20], that is about helping end users to manage cloud systems dynamically. It introduces a cloud intention layer that separates the cloud source code and services and helps the software application to adopt user's requirements at runtime. Many other user-focuused service-oriented applications such as [21] and [22] have also been proposed in the literature. Thus, cloud service engineering has opened new doors for applications to provide services to the end users. Similarly, in our approach, we are building a service-oriented application to enable sharing on top of the Amazon EC2 infrastructure. Service cost is not associated but the value is given based on the trust of social networks.

As compared to existing application-specific models, the CRB model is more generic and flexible. In future, it can be extended to various social platforms for sharing cloud resources. It allows communities to make their own set of policies and supports an open and ad-hoc bartering model for sharing resources. Our strategy may reduce the costs associated with assessing cloud services and may also encourage resources to be utilized well. In the literature, social models are found, that are adopting similar approaches [23] but we aim to deliver a flexible system that is specifically designed for the needs of those in developing countries.

3. METHODOLOGY

It has been observed from the literature review that a resource bartering strategy may help social communities to share and save money, as bartering is a mechanism that does not involve direct costs. For this purpose, we describe a new container based cloud resource bartering (CRB) model which is explained in this section. This can be thought of as a sub-letting process in which the landlord user rents computing resources from the cloud provider and then sublets excess capacity to tenants who in turn agree to abide by a specific bartered agreement. Figure 1 shows high-level service-oriented architecture that supports cloud resource bartering among Facebook users. The system eases cross-domain capabilities integration for flexible cost efficient and trust-based sharing of container-based virtual instances. A cloud Resource bartering (CRB) application is hosted by an application server.

We targeted a commercial Amazon EC2 platform for seamless integration with proposed CRB model because of its wide availability and stability level. Figure 1 shows the layered architecture of cloud resource bartering (CRB).

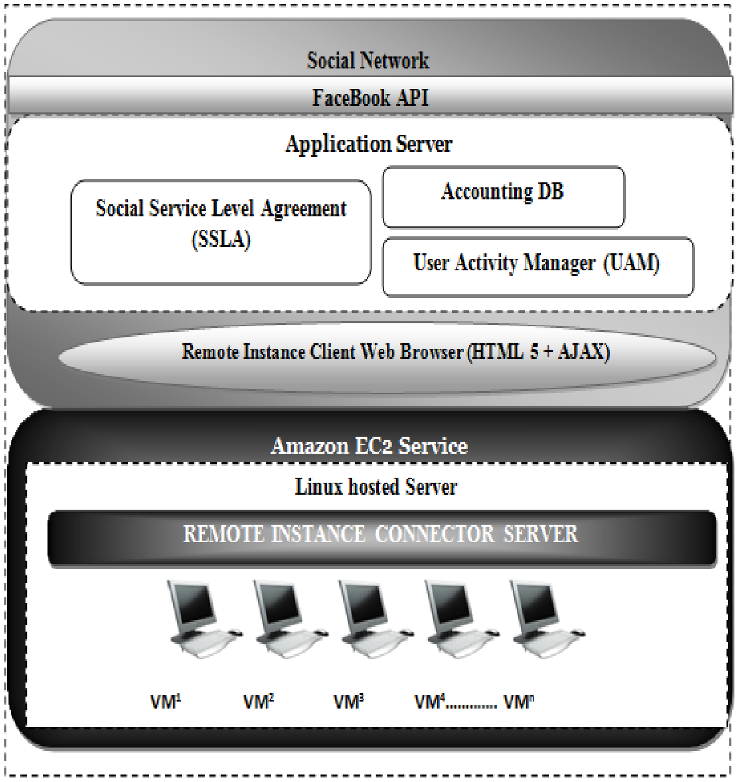

Figure 1. Layered architecture of CRB-model

In Figure 1, the first layer represents the Facebook API which provides the community with a means of communication through which they can share their Amazon EC2 instances over the internet. The landlord is able to negotiate with, manage and select their tenants using this social context.

The second layer represents the CRB Application that has three submodules. The User Activity Manager (UAM) is one sub-module of the CRB application. It shows the log of the shared activities of the community, and it also keeps a track of landlord users and their corresponding shared virtual machines details. The Social Service Level Agreement SSLA is another submodule of CRB application and it is offered by a resource owner (landlord) which is agreed with the tenant before they may share a resource. Via the creation of a SSLA agreement, the landlord can release resources to their tenants. The third layer shown in Figure 1 assists tenants in accessing EC2 instances remotely. The CRB application is a client that makes a request for establishing a remote connection with an Amazon instance. After verifying a client's credentials, a remote desktop web service allows access to EC2 instance or container-based virtual instance which results in successful resource sharing of CRB application. In the following section, the functions of cloud recourse bartering are explained in detail.

3.1 Functions of Cloud Resource Bartering (CRB)

The proposed system consists of two main functionalities. The first One of these is the user registration process which will be discussed in the following section.

1) The Landlord User Registration Process

The workflow for sharing cloud resources is explained in figure 2. Facebook provides an interface to the landlord for registering the resources that will be available for bartering. The owner's information and shared resource details are recorded in a database by the application server.

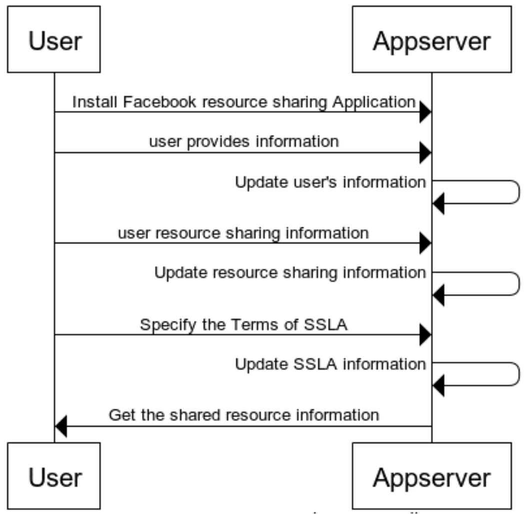

Figure 2. User's registration process

The registration process provides the following functionalities in terms of registering a resource. A landlord can state the terms and conditions for sharing a resource in (SSLA).

- The landlord can then share a resource in exchange for a favor from the tenant which satisfies the bartering mechanism.
- A user activity manager (UAM) will ensure that all tenants are following a suitably signed SSLA.
- The landlord may also share resource voluntarily without requiring the signing of a SSLA.

Would-be tenants can view the resources that are available and make requests for them only after successfully completing the registration process. A landlord has the right to grant or to take the shared instance back at any time, although a warning period should be given to tenants before access is revoked.

2) The Resource Sharing Process

The tenant will install the Facebook application in order to begin resource bartering as shown in following figure 3.

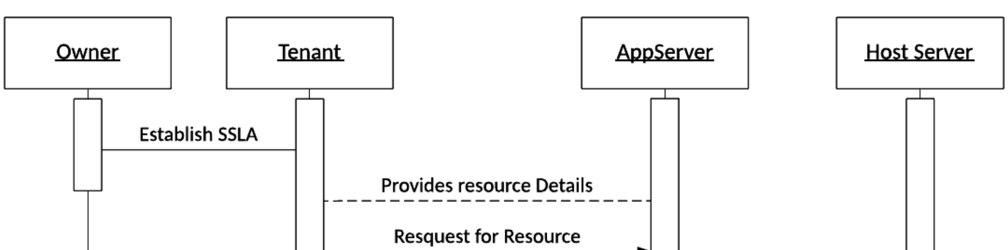

Figure 3. Resource sharing process with a user

- The application server provides a list of shared EC2 instances to the prospective tenant.
- A user can request access to a resource by establishing SSLA with the landlord of a given virtual machine.
- The application server establishes a remote connection with the Amazon EC2 server to share the complete instance or an LXC of that instance.
- The landlord keeps the administrative rights to grant access to the resource or to end the sharing agreement at any time.
- LXC containerization is used to keep the tenant's virtual machine usage separate from that of the landlord and other tenants and to manage the resources that they consume. This provides a reasonable level of security for all the participants in the system whilst at the same time being light-weight enough not to seriously affect system performance.

3.2 Implementation Details:

The implemented CRB application is an Active Server Pages (ASP) based web application that provides an interface to users through which they can trade resources. The bartering application is integrated with Facebook to encourage sharing between social community members. A Guacamole web service is embedded within the CRB application to make a remote connection with Amazon EC2. An LXC container is installed in the Amazon EC2 instance to allow users to benefit from container-based virtualization. The following Figure 4 provides an overview of implemented CRB application. This application keeps a track of complete details of user's trading activities, for this purpose a complete list of resources that a user has shared is maintained. The record of resources acquired by a user from others is also maintained. The CRB application also provides details about Amazon EC2 instance which includes the instance type, RAM, and state of instance whether it is running or terminated.

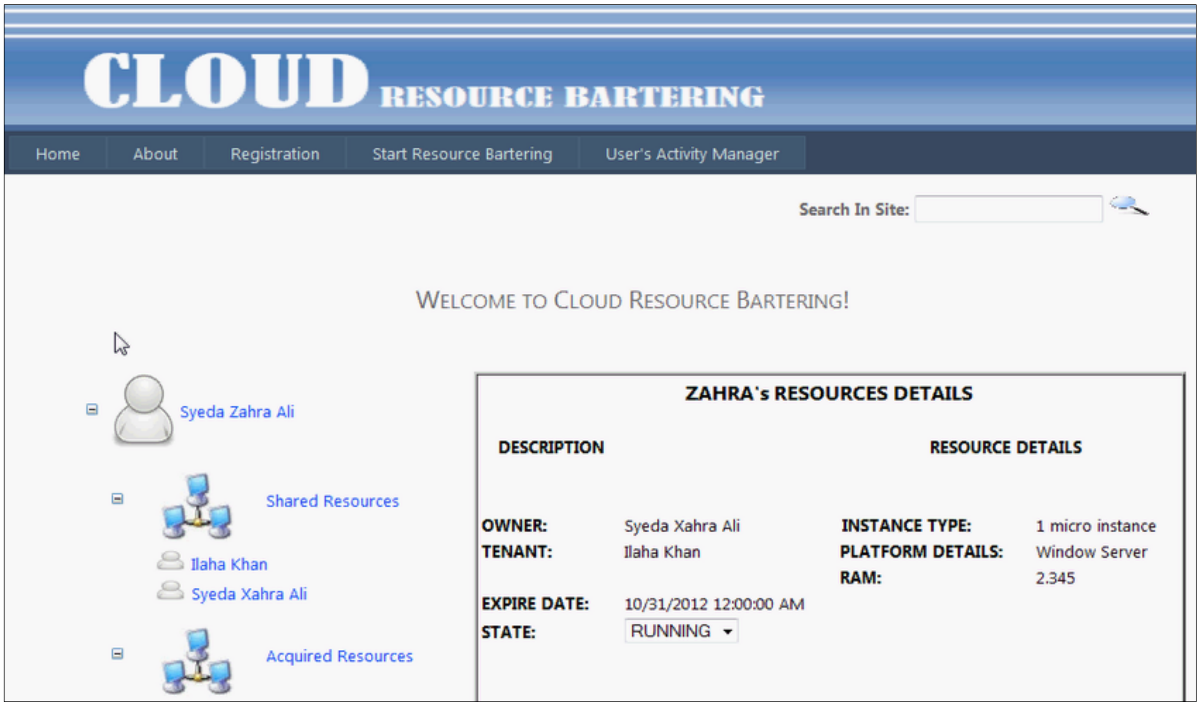

Figure 4 Resource bartering marketplace

A tenant can access a cloud resource by requesting an account from its owner. Before a tenant can acquire a cloud resource they need to accept a social service level agreement (SSLA). Following this the bartering application establishes a connection with the Amazon remote instance and grants access to the tenant. The owner can share the same instance with multiple friends at the same time with the help of implemented container-based virtualization (LXC).

The user activity manager (UAM) monitors all the behavior of users. It keeps track of all the SSLA's and provides details regarding whether a tenant has fulfilled all the terms and conditions of their SLA's or has violated them. The user activity manager (UAM) thereby helps the user to view the complete status of resource bartering and their respective details.

4. Evaluation

Based on the inspiration from the literature, the testing of the proposed model has been carried out in two phases. Evaluation Phase I collects qualitative feedback from the users, which is then used in order to determine the usefulness of CRB model for a social community in a practical situation. Evaluation Phase II presents the quantitative analysis of CRB model. In this phase, the performance of the cloud resources is tested within a shared community environment. We believe that together these two phases will provide evidence that can be used to understand the effectiveness of the model.

4.1 Evaluation Phase I: Experimental Results and Analysis

The CRB-Model focuses on the social aspect of resource sharing and as such the social community plays a central role in evaluating its performance. To gather practical feedback regarding this aspect of the system, a qualitative testing technique has been adopted. Common feedback capturing techniques include contextual inquiry, interviews with a focus group, questionnaires, surveys, self-reporting logs and journalized sessions. Such methods collect information from the users while they use the system in question. Among all these feedback choices, a questionnaire was felt to be the most appropriate option as it provides a standardized and relatively simple way to collect information in a structured way. Our study was designed for basic statistical analysis to ascertain how a community of users might feel about using the system. A group of Facebook users was chosen to execute a real-time resource bartering scenario using the system. The targeted group used the CRB-Model to share and acquire Cloud resources via a social network. Their feedback was then recorded using a web-based questionnaire. The sample data[4] from this survey and its analysis[5] is available on the given links in footnotes.

The feedback regarding the CRB-Model is assessed using different evaluation parameters. The results highlight that the major part of the community successfully grasped the concept and understood the system. It also indicates that the CRB-Model is deployed on a well-recognized social platform and users are already familiar with this environment. It was observed that all users are familiar with Facebook and a major part of community easily adopted the appearance of the system. In this context the questionnaire feedback taken from a social group is acceptable. This is because they are already familiar with the Facebook interface, so they did not find it difficult to learn the CRB application interface. Furthermore, it has been seen that pre-built trust relationships in Facebook encourage users to share their resources with their friends. For this purpose, the feedback of social community seems acceptable. This testing approach presented a detailed feedback taken from the community-related to the proposed CRB-model. The overall opinion was quite positive regarding the system. Users easily get started with the application as it was deployed on a social network. By analyzing the feedback from the social community, it was found that the CRB application provides a convenient way of sharing Amazon EC2 instances with other trusted users. Details regarding the second evaluation phase and the results from it are presented in the next section.

4.2 Evaluation Phase II:

The focus of evaluation phase II is to determine the feasibility of CRB application in different workload scenarios. It is important to measure the Amazon EC2 instance performance relative to the number of users accessing it through containers.

---

[4]https://docs.google.com/a/seecs.edu.pk/spreadsheet/ccc?key=0Am4V3taJ3KwLdE9xb1NDRW9mSjNkSkpUTHdjVTd1Mmc#gid=0
[5]http://apps.facebook.com/339352909494540/?fb_source=bookmark_apps&ref=bookmarks&count=0&fb_bmpos=3_0

The CRB-Model enables sharing the Amazon EC2 virtual machines (from here on 'host'); hence the goal in this phase is to analyze the performance impact on shared containers when the host runs different heavy workloads. We also study the performance impact on other containers as well as on the host machine, when only some containers are heavily loaded. A range of testing techniques could be adopted for this purpose, however, two different scenarios have been designed with various test cases, to evaluate the relative performance of host and containers under different conditions. In the following section, the experimental setup and quantitative testing tools are explained in detail for system assessments. Consequently, conclusions are drawn from the results.

4.2.1 Experimental Design:

In this testing approach, an Amazon EC2 M1 large instance was used with 7.5 GB memory, 4 EC2 Compute Units (2 virtual cores with 2 EC2 Compute Units each), 850 GB instance storage and a 64-bit platform. For the M1 large instance, I/O performance is high. A performance analysis tool was required to monitor the host virtual machine's performance relative the containers running in it and vice-versa, with a variety of workloads and scenarios. There is a range of testing tools available for this purpose. These include Copperegg[6], Amazon CloudWatch [24], RevealCloud for Amazon EC2 monitoring and the Free EC2 Health Monitor [25]. The Phoronix[7] testing tool was chosen for this evaluation due to its robust analysis and open benchmarking aspect. This open benchmarking testing suite was installed in the Amazon EC2 large instance in order to analyze its performance under different workload conditions. A number of nested LXC instances were created in the Amazon EC2 large instance and configured to run Phoronix.

A stress[8] package was installed in the host as well as in the containers. It enabled a configurable I/O, CPU and disk workload to be placed on the systems. Phoronix test suite supports a number of performance tests such as PyBench[9]. It presents a standardized way to measure the overall performance of the system and is also used to see the performance of host machine and of Linux containers installed in it.

4.2.2 Design of the Test case scenario:

Two scenarios were tested under the environment described previously. While designing the test cases, the focus was to monitor the performance of the Amazon EC2 host machines and Linux containers installed in it. Thus, the goal of this evaluation phase was to understand how CRB Model-based resource sharing in the social community affects the performance

---

[6] http://copperegg.com
[7] http://www.phoronix-test-suite.com/
[8] http://www.stresslinux.org/sl/
[9]http://openbenchmarking.org/test/pts/pybench

of host and containers, so two test scenarios were designed. In the first scenario, the host machine is put under different workload conditions, and the effect is observed on the containers running in it. In the second scenario, the performance of the host machine is measured with reference to nested machines by executing test PyBench in nested machines. PyBench categorizes and mentions different workloads with the terminology of Workload 0, 1, 8 and so on. Workload 0 means a workload with no stress or heavy workload, and Workload 1 comes with a certain stress weight.

4.2.3 Scenario 1:

Five Linux containers named C0, C1, C2, C3, and C4 were created in an Amazon EC2 large instance. Table 1 shows that C0 runs a task in just 25% of the time when the host machine was running Workload 0, as compared to when it was running Workload 64. Similar is the case with other containers. Hence it can be deduced that the performance of containers/ nested machines is relative to the performance of the host machine. The average task execution time for all containers with no workload is 4301 ms and with a workload 64, it increases to 16875 ms. This indicates that the overall performance of the host machine decreases by a quarter.

Table 1. Performance measurement of containers, with host machine under various workload

Host	***C0***	***C1***	***C2***	***C3***	***C4***
Workload 0	**4301 ms**	**4284 ms**	**4299 ms**	**4309 ms**	**4290 ms**
Workload 1	**4291 ms**	**4303 ms**	**4299 ms**	**4294 ms**	**4307 ms**
Workload 6	**5802 ms**	**5811 ms**	**5710 ms**	**5818 ms**	**5814 ms**
Workload 8	**7950 ms**	**8014 ms**	**10083 ms**	**10053 ms**	**9995 ms**
Workload 16	**12193 ms**	**12303 ms**	**12346 ms**	**12399 ms**	**12419 ms**
Workload 32	**14700 ms**	**14463 ms**	**14545 ms**	**14640 ms**	**14776 ms**
Workload 64	**16875 ms**	**16881 ms**	**17132 ms**	**16841 ms**	**16750 ms**

Finally, a comparison of all nested instances' PyBench task execution time obtained by each test run is shown in Figure 5. The graph shows that the performance of the host machine is directly proportional to the performance of the nested instances and inversely proportional to the workload weight added to the system. The figure 5 also highlights that the trend of task execution time is almost same for all the containers and it heavily depends on the host machine's resource utilization. Therefore, we can conclude that workload weight on the host machine affects all the users sharing it through containers.

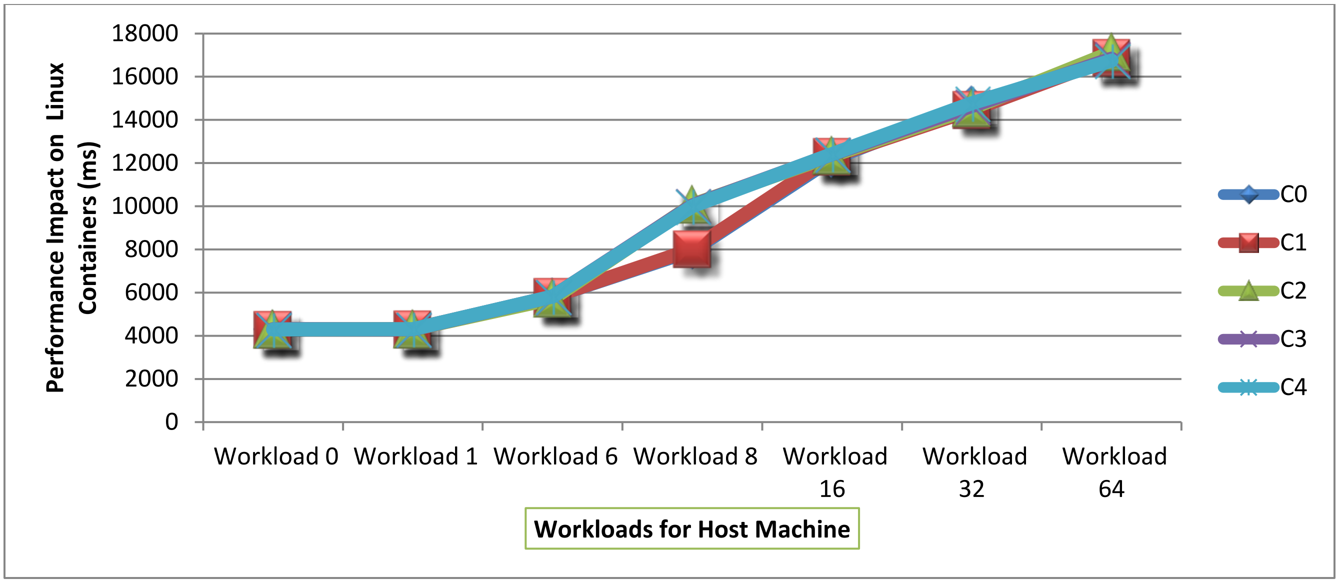

Figure 5. Observing the performance impact on containers by stressing the host machine

4.2.4 Scenario 2:

In this scenario, the host machine and the containers/ nested machines are evaluated under different workload conditions. The focus here is to find out the effect on host and containers when some of them run the heavy workload. The following table 2 gives an insight that the performance of the host machine does not vary too much whether only two containers (C0 and C1) are heavily loaded or all of them. It is quite obvious that the stressed containers take more time to execute PyBench, however, the containers under stress do not significantly affect the performance of other containers. The nested containers under stress have a local performance decrease when compared to the other containers with no stress.

Table 2. PyBench task execution times of host and containers with different workload weights

	C0,C1 under stress	***C0,C1,C2 under stress***	***C0,C1,C2,C3 under stress***	***C0,C1,C2,C3,C4 under stress***
Host	**3570 ms**	**3570 ms**	**3590 ms**	**3569 ms**
C0	**4640 ms**	**8883 ms**	**11748 ms**	**15946 ms**
C1	**4717 ms**	**8858 ms**	**11688 ms**	**16441 ms**
C2	**3838 ms**	**8971 ms**	**11559 ms**	**16056 ms**
C3	**3823 ms**	**4679 ms**	**11598 ms**	**16293 ms**
C4	**3826 ms**	**4742 ms**	**5513 ms**	**16222 ms**

Figure 6 gives a graphical depiction of the results and highlights that the performance of the host machine stays almost same irrespective of the load in the nested containers. It also explains that for the test run 1, C0 and C1 take a little more time in executing the PyBench task, as compared to C2, C3, and C4. In a test run 2, the performance of C3 and C4 remained almost unchanged, while the stressed containers performed poorly. It shows that the stressed containers do not affect the performance of the other containers. A similar trend can be noticed in a test run 3. In a test run 4, all the containers performed equally while the host machine's performance remained unchanged. It shows that even if all the containers are heavily stressed, the host machine will still be able to perform reliably well. The results support our claim that through CRB model the cloud resources (i.e. virtual machines) can be effectively shared with other users.

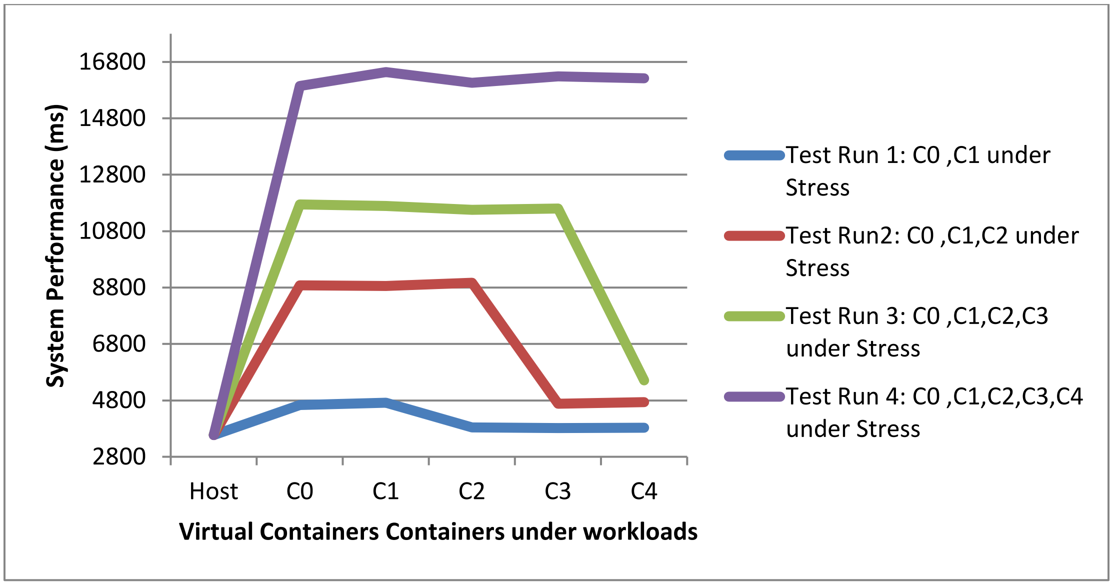

Figure 6. The performance impact on host machine and containers by stressing a few of them

4.2.5 Result Analysis:

The results show that although the performance of a container may decrease due to a heavy workload, it has a very small impact on the other nested instances. The following figure 7 gives consolidated results to show that the performance of the host machine stays same no matter how heavily the containers are loaded. It also shows that the containers under stress do not affect the performance of the other containers running on the same host. Hence it can be concluded that the CRB system performed effectively well under different workloads, and demonstrated its feasibility. The results provide evidence that resource sharing between multiple users works effectively after deploying the CRB Model. The environment set up with containers helps to insulate users from each other's workloads. This confirms that resources can be effectively and fairly shared between users through the CRB application.

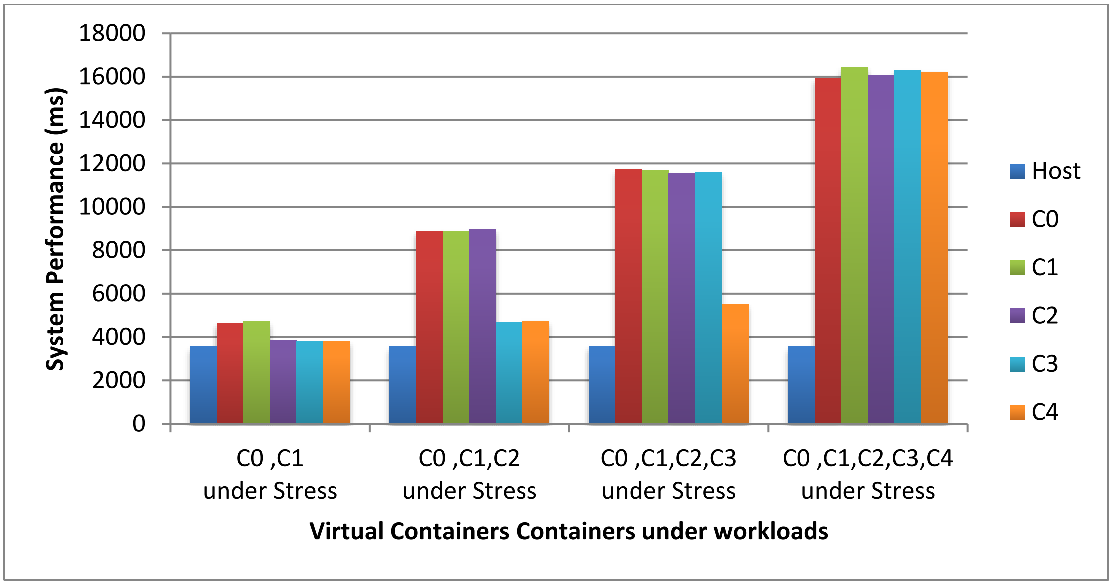

Figure 7. Systems' performance with reference to nested Linux containers with different workload weights

The performance of the prototype has been assessed during two evaluation phases. In the first stage, feedback was gathered from a group of end users within a social community. This confirmed that existing social networks can provide a solid trust foundation for the sharing of Cloud resources. This analysis gave evidence that in developing countries, especially within the research or student communities; users can effectively utilize the Cloud resources by adopting this kind of model. The evaluation was extended by a quantitative approach to observing the performance of the Amazon EC2 instance under various sharing environments. In evaluation phase - II it was found that multiple users can share Amazon EC2 resources using the CRB-Model. The impact of sharing resources was evaluated, and conclusions were made regarding this aspect of the system. Results demonstrated that the EC2 instances kept stable and worked effectively under a range of different test conditions and sharing scenarios.

5. Conclusion and future work

In developing countries, it is often too expensive for people to acquire a virtual machine from public clouds. Users may, therefore, wish to manage costs and increase computational resource usage by sharing the virtual instances with others. This work provides a Cloud resource bartering platform to help users share their cloud resources through Facebook-based social network, just like they would share other content. Based on this idea, this work presents a system to help users to share their dedicated resources without the need for money changing hands in different social communities, in a fair trust-based environment. The results provide a clear confirmation that under different workload conditions, cloud resources can be effectively shared within a social community. This indicates the successful resource sharing using the proposed CRB Model and confirms that the prototype shows great promise in terms of helping users in developing countries access resources which they couldn't otherwise afford. This may help users to save money whilst accessing the latest technology.

Cloud resource sharing in a social network does bring some security considerations. In future, we aim to propose several different security techniques to make sharing more secure using a social network. Furthermore, the presented CRB model is set to be extended for other social networks (such as Google+, Twitter etc.) to enable cloud resources to be shared between users. In future, we intend to deploy CRB in a generic social environment such as OpenSocial. The CRB-Model can be extended for a variety of applications. This might include, for example, helping the scientific community to manage and share their computing resources. A large social group can work on a single project simultaneously using the CRB-Model which might be of benefit to the open source community.

Acknowledgment

We would like to express our gratitude to Amazon EC2 for providing us a research grant making it possible to test and evaluate the proposed economic model in a real-time cloud trading scenario. It provided us with a high-performance computing environment allowing us to tap into the high demand infrastructure of AWS cloud.

REFERENCES

[1] Goble CA, Bhagat J, Aleksejevs S, Cruickshank D, Michaelides D, Newman D, et al. myExperiment: A repository and social network for the sharing of bioinformatics workflows. Nucleic Acids Res 2010;38. doi:10.1093/nar/gkq429.

[2] Ludäscher B, Altintas I, Berkley C, Higgins D, Jaeger E, Jones M, et al. Scientific workflow management and the Kepler system. Concurr Comput Pract Exp 2006;18:1039–65. doi:10.1002/cpe.994.

[3] Cock PJA, Grüning BA, Paszkiewicz K, Pritchard L. Galaxy tools and workflows for sequence analysis with applications in molecular plant pathology. PeerJ 2013;1:e167. doi:10.7717/peerj.167.

[4] Guo Z, Singh R, Pierce M. Building the PolarGrid portal using web 2.0 and OpenSocial. Proc 5th Grid Comput Environ Work - GCE '09 2009:1. doi:10.1145/1658260.1658267.

[5] Mitchell-Wong J, Kowalczyk R, Roshelova A, Joy B, Tsai H. OpenSocial: From social networks to social ecosystem. Proc. 2007 Inaug. IEEE-IES Digit. Ecosyst. Technol. Conf. DEST 2007, 2007, p. 361–6. doi:10.1109/DEST.2007.371999.

[6] Singh MP, Huhns MN. Automating workflows for service order processing: Integrating AI and database technologies. IEEE Expert 1994;9:19–23. doi:10.1109/64.331480.

[7] Chard K, Caton S, Rana O, Bubendorfer K. Social Cloud: Cloud computing in social networks. Proc. - 2010 IEEE 3rd Int. Conf. Cloud Comput. CLOUD 2010, 2010, p. 99–106. doi:10.1109/CLOUD.2010.28.

[8] Thaufeeg AM, Bubendorfer K, Chard K. Collaborative eResearch in a social cloud. Proc. - 2011 7th IEEE Int. Conf. eScience, eScience 2011, 2011, p. 224–31. doi:10.1109/eScience.2011.39.

[9] Goiri Í, Guitart J, Torres J. Economic model of a Cloud provider operating in a federated Cloud. Inf Syst Front 2012;14:827–43. doi:10.1007/s10796-011-9325-x.

[10] Belalem G, Bouamama S, Sekhri L. An effective economic management of resources in cloud computing. J Comput 2011;6:404–11. doi:10.4304/jcp.6.3.404-411.

[11] Ali Z, Rasool RU, Bloodsworth P. Social networking for sharing cloud resources. Proc. - 2nd Int. Conf. Cloud Green Comput. 2nd Int. Conf. Soc. Comput. Its Appl. CGC/SCA 2012, 2012, p. 160–6. doi:10.1109/CGC.2012.37.

[12] Abdallah A, Shen X. Lightweight security and privacy preserving scheme for smart grid customer-side networks. IEEE Trans. Smart Grid, vol. PP, 2015. doi:10.1109/TSG.2015.2463742.

[13] Neglia G, Reina G, Zhang H, Towsley D, Venkataramani A, Danaher J. Availability in BitTorrent systems. Proc. - IEEE INFOCOM, 2007, p. 2216–24. doi:10.1109/INFCOM.2007.256.

[14] Anderson DP, Cobb J, Korpela E, Lebofsky M, Werthimer D. SETI@home: an experiment in public-resource

computing. Commun ACM 2002;45:56–61. doi:10.1145/581571.581573.
[15] Andrade N, Cirne W, Brasileiro F, Roisenberg P. OurGrid: An Approach to Easily Assemble Grids with Equitable Resource Sharing. Job Sched Strateg Parallel Process 2003;2862:61–86. doi:10.1007/10968987_4.
[16] Buyya R, Abramson D, Giddy J, Stockinger H. Economic models for resource management and scheduling in Grid computing. 2002. doi:10.1002/cpe.690.
[17] Buyya R, Venugopal S. Cluster and grid computing: A graduate distributed-computing course. IEEE Distrib Syst Online 2007;8. doi:10.1109/MDSO.2007.4415516.
[18] Baker T, Mackay M, Randles M, Taleb-Bendiab A. Intention-oriented programming support for runtime adaptive autonomic cloud-based applications. Comput Electr Eng 2013;39:2400–12. doi:10.1016/j.compeleceng.2013.04.019.
[19] Tai S, Nimis J, Lenk A, Klems M. Cloud service engineering. Proc 32nd ACM/IEEE Int Conf Softw Eng - ICSE '10 2010;2:475. doi:10.1145/1810295.1810437.
[20] Baker T, Mackay M, Randles M. Eternal cloud computation application development. Proc. - 4th Int. Conf. Dev. eSystems Eng. DeSE 2011, 2011, p. 392–7. doi:10.1109/DeSE.2011.70.
[21] Baker T, Taleb-Bendiab A, Al-Jumeily D, Miseldine P. Creating on-demand service-oriented applications from intentions model. Proc. - Int. Conf. Dev. eSystems Eng. DeSE 2009, 2009, p. 219–26. doi:10.1109/DeSE.2009.40.
[22] Taleb-Bendiab A, Miseldine P, Randles M, Baker T. Programming support and governance for process-oriented software autonomy. Lect Notes Bus Inf Process 2008;8:3–18. doi:10.1007/978-3-540-68262-2_1.
[23] Zhang Q, Cheng L, Boutaba R. Cloud computing: State-of-the-art and research challenges. J Internet Serv Appl 2010;1:7–18. doi:10.1007/s13174-010-0007-6.
[24] © 2017, Amazon Web Services I or its affiliates. A rights reserved. Amazon CloudWatch 2017. https://aws.amazon.com/cloudwatch/.
[25] Reserved. ZCA rights. Free EC2 Health Monitor 2017. https://www.manageengine.com/free-ec2-health-monitor-tool/free-ec2-health-monitor-index.html.

Ms. Zahra Ali

Zahra Ali completed her M.Sc. degree in Information Technology from NUST School of Electrical Engineering and Computer Science (NUST-SEECS), Pakistan, in 2013. Her main research interests are Distributed Computing and Social Computing.

Raihan ur Rasool

Dr. Raihan ur Rasool is a Fulbright Alumnus of the University of Chicago, USA. He is currently affiliated with Victoria University, Melbourne. His research interests include large-scale systems, security and computer architecture. His research work, comprising over 45 articles is published in various international conferences and journals, e.g. ISCA, HiPEC, CCGrid, ACM SIGARCH, IEEE Transactions on Cloud Computing, JNCA & FGCS.

Shamyl Bin Mansoor

Shamyl Bin Mansoor is an Assistant Professor at School of Electrical Engineering and Computer Science at National University of Sciences and Technology. His areas of interests include educational technology, internet of things, modeling and simulation. Along with his research interests and teaching he runs a successful educational technology company.

Peter Bloodsworth

Dr. Peter Bloodsworth joined the Department of Computer Science at the University of Oxford in 2017. Prior to this he worked and lived in Islamabad, Pakistan at the National University of Sciences and Technology (NUST) as a Foreign Professor from

autumn 2011 - 2016. He has a Ph.D. in Multi-agent Systems from Oxford Brookes University.