Autonomous, Seamless and Resilience Carrier Cloud Brokerage Solution for Business Contingencies during Disaster Recovery

Sonia Shahzadi*, George Ubakanma†, Muddesar Iqbal‡, Tasos Dagiuklas§
*Email: s.shahzadi@swanmesh.com † ‡§Email: {ubakang, m.iqbal, tdagiuklas}@lsbu.ac.uk
* Swan Mesh Networks Ltd, Research and Development, London, UK
† ‡§School of Engineering, London South Bank University, UK

***Abstract*—The challenge of disaster recovery management for cloud based services is constantly evolving. The costs of cloud service downtime in the event of disaster striking is the subject of much international research. The key issue to resolve is developing suitably resilient and seamless live/real-time mechanisms for disaster recovery. In this paper, we have implemented a proof of concept for an autonomous and fault tolerant carrier cloud brokerage solution with resilient provisioning of on-the-fly cloud resources. When a disaster strikes, the proposed solution will trigger the migration of an entire IaaS from one cloud to another without causing any disruption to the business. In the event of non-availability of hosts for the deployment of virtual network functions for different business processes, an on-the-fly host selection mechanism is proposed and implemented to locate other active compute hosts without any disruptions. In order to evaluate the performance of the proposed solution, we defined several use-case scenarios for each cloud service. This proposed solution will not only reduce the capital expenditure but also provides a reliable and efficient way to access the data during disaster.**

***Keywords*-Cloud Computing, Business Continuity, Infrastructure as a Service, Platform as a Service, Software as a Service**

I. Introduction

Disaster recovery in cloud computing has gained a lot of attention due to its benefits that facilitate the needs of business. Currently, many business organizations face diverse disruptions that could affect organizational assets. Therefore, a proactive approach is required in order to counteract these disruptions. Mostly, these organizations depend on Disaster Recovery (DR) services to prevent service disruptions, because even short periods of downtime can cause significant business losses [1]. Generally, these DR services are expensive increasing the organizations capital costs. The proposed solution addresses on-demand and autonomous mechanisms to assure high availability of services in disaster situations. When we study disaster recovery in cloud environments different options are available. Cloud computing depends on cloud models i.e. private cloud, public cloud and hybrid cloud [2].

The Sendai Framework for Disaster Risk Reduction 2015-2030 [3], was adopted at the Third UN World Conference in Sendai, Japan, in March 2015. The Sendai Framework outlines the need to increase investment efforts in Disaster Reduction for Resilience. As part of this on-going research effort we have developed a cost effective, fault tolerant and resilient carrier cloud architecture that can be deployed at short notice in disaster management and recovery situations. The solution builds on the principles of live migration and disaster recovery [2]. In order to keep the solution flexible and accessible open source technologies and existing standards are utilized to maximum effect in delivering a practical, robust and extensible solution.

Cloud computing provides feasible disaster recovery solutions due to its dynamic scalable and high availability structure. To reduce the disaster outcomes, a multi-cloud disaster recovery model is implemented that mange the resources from multiple cloud providers. In this paper, we will argue that cloud computing is an effective platform for DR services with low costs, and it minimizes recovery time without data loss. The reason to choose cloud based DR solution [4]:

- Easy to deploy and manage
- Maximum flexibility
- Agility
- High availability
- Reduce capital cost
- Reduce DR solution cost
- Ease of access for business continuity

The remainder of this paper is organized as follows. Section II describes the relevant work and state of the art. Section III presents use cases relevant to cloud disaster management. Section IV presents the architecture of our proposed solution. Section V describes the implementation of a proposed framework as proof of concept. Section VI describes the performance evaluation while section VII describes the conclusion of this paper and future work.

II. Relevant work

Today, as technology evolves rapidly, dynamic and responsive applications are required to support users. Development flow of Business Continuity (BC) and disaster recovery are given in [5] where two parts of a disaster recovery program (i.e. building the program and choosing cloud solution) using cloud are discussed. According to study

[4], many organizations are considering migration to cloud computing for business continuity and a survey revealed that only 50% of re-respondents have proper DR plans for the whole organization, while 36% say they only have DR plans for back-end infrastructure that will only work for data center, not for remote offices and desktops, 7% don't have DR plans, but they can deploy within 6 months, 5% don't have DR plans, but they can deploy within 12 months and 2% also don't have DR plans, but they can deploy within 24 months. Business Continuity and Disaster Recovery (BCDR) plans for healthcare scenario are discussed in [6] to make sure fast and secure access of patient data with reasonable budget. A practical solution is implemented to deal with disaster recovery and business continuity[7] in Portugal.
Cloud brokerage solutions can provide on-the-fly configuration of existing cloud platforms at the Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) levels [8], while supporting portability and interoperability solutions [9]. Mostly, the cloud portability and interoperability approaches are put into practice to overcome certain cloud platform limitations. Although, portability and interoperability technically complement each other, both approaches differ in many ways [9]. To reduce the human effort, design and deployment of cloud applications is provisioned such that solutions can move from one cloud to another, this is called portability [9]. Whereas cloud interoperability supports different cloud services and applications to work together and the customer is largely unaware of this [9]. Based on the literature review, the existing approaches can be categorized into the following two units [9]:

A. Using existing standards

1) Cloud Data Management Interface (CDMI) specifies how applications: create, delete, update and retrieve data on the cloud.
2) Open Cloud Computing Interface (OCCI) supports the deployment, monitoring and autonomic scaling. Its API supports compute, network and storage services [9].
3) Open Virtualization Format (OVF) for import and export of VMs using OVF standards.
4) Topology and Orchestration Specification for Cloud Applications (TOSCA) specify a language to a define service and its components to deal with portability [8, 9].

B. Using open source libraries

1) Jclouds abstract the differences between multiple cloud providers and also provide application portability [10].
2) δ-Cloud is REST based API framework in Ruby language that abstract the differences between multiple cloud IaaS platforms [11].
3) Libcloud is a Python library that support extensive cloud providers APIs [12].
4) Fog provides a high level interface to different clouds using ruby language [13].
5) Dasein Cloud is a Java based library for compute service access [14].
6) Simple Cloud is a PHP based library for storage, queue and infrastructure services [15].

These open source abstraction solutions provide an intermediate layer for cloud management [8, 9, 16, 17, 18] while comparisons of these solutions are given in Table I.

III. Cloud Disaster Management: Use Cases

Regardless of whether the disaster is classified as:
Natural e.g. hurricane, tornado, flood or earthquake.
Man-made e.g. infrastructure failure and cyber-attacks etc.
A Disaster Recovery (DR) plan incorporating procedures and techniques for prevention, mitigation, managing recovery is essential. Within the scope of the practical measures for executing the DR plan, our research focuses on the delivery of an instant and scalable approach enabling:

A. Reduced Recovery Waiting Times

For good DR services according to cost, these matrices are used: Recovery Time Objective (RTO), Recovery Point Objective (RPO), performance and geographic separation [19]. We are developing a proof of concept utilizing our Multi-Cloud Broker Orchestrator and Cloud Carrier Architecture to provision instant (on the fly) deployment/restoration of essential data center services. This enables deployment of essential services sooner when a disaster has struck. To enable this were centrally orchestrating all services instead of provisioning them separately. Our goal is improving metrics such as RTO, the maximum time to service recovery and the RPO the maximum allowable data loss. The values assigned to metrics will be defined on an individual basis by problem domain and application [20]. Cloud Replication backup approach offers short Recovery Time Objectives (RTOs) with maximum protection for critical applications [6].

B. Service Traffic Optimization

Our goal is to optimize traffic flows to help keep recovery time and data losses down, while minimizing costs. Our concept employs open source Apache libcloud to utilize a wide range of cloud resources and to optimize in real-time, both inter-cloud management and load balancing.

C. Backup as a Service

For good DR services according to backup, these matrices are used: Hot Backup Site, Warm Backup Site and Cold Backup Site [19]. Multi-Cloud Broker Orchestration and Cloud Carrier Architectures can play an increasing role in delivering Backup as a Service. Ensuring that clients can not

Table I
COMPARISON OF CLOUD ABSTRACTION SOLUTIONS

Features	Jclouds	δ-Cloud	Libcloud	Fog	Dasein Cloud	Simple Cloud
Type	Library	Framework	Library	Library	Library	Library
Database	*N*	*N/A*	*N*	*N*	*N/A*	*N/A*
Supported IaaS	*Y*	*Y*	*Y*	*Y*	*Y*	*Y*
Supported PaaS	*Y*	*Y*	*Y*	*N/A*	*N/A*	*N/A*
Multi-IaaS Support	*Y*	*Y*	*Y*	*N/A*	*N/A*	*Y*
Multi-Cloud Support	*Y*	*Y*	*Y*	*N/A*	*Y*	*Y*
DNS	*N*	*N/A*	*Y*	*Y*	*N/A*	*N/A*
License	Apache License 2.0	Apache License 2.0	Apache License 2.0	MIT License	Apache License 2.0	Open BSD License
Documentation	Good	Good	Good	*N/A*	Very less to no documentation.	Good
Supported CSPs	30	17	60	43	*N/A*	*N/A*
Compute	*Y*	*Y*	*Y*	*Y*	*Y*	*N/A*
Network	*N*	*Y*	*Y*	*N/A*	*Y*	*N/A*
Storage	*Y*	*Y*	*Y*	*Y*	*Y*	*Y*
Amazon EC2 Support	*Y*	*Y*	*Y*	*N*	*Y*	*Y*
Programming Language	Java	Ruby	Python	Ruby	Java	PHP
Platform Integration	Maven	Drivers	Drivers	*N/A*	*N/A*	*N/A*
EBS Storage Support	*Y*	No, but available in road-map.	*N*	*N/A*	*N/A*	*Y*
Container	*N*	*N/A*	*Y*	*N*	*N/A*	*N/A*
CDN	*N*	*N/A*	*Y*	*Y*	*N/A*	*N/A*
Load Balancing	*Y*	*Y*	*Y*	*N*	*N/A*	*N/A*

Y=Yes; N=No; N/A=Not Available;

only recover their cloud infrastructure and services on the fly when disaster strikes; but also be assured of reliable access to backup data resources as vital services are re-stored [21]. Based on the use-cases discussed above, to provision cloud contingency services during emergency and dis-aster situations, we consider following solutions to deliver the cloud infrastructure as a service.

During a natural disaster when communication infrastructure is destroyed, it can also effect regional cloud datacenters located within that specific zone. This can partially or fully disrupt business services, resulting in potential losses of millions. The Cloud is a promising solution to providing on-demand provisioning services between different regions. As a cloud manages all these resources through a central location. Therefore, as shown in Figure 1, if regional a cloud goes down or collapses for example in Zone B, then all the user requests will be redirected to Zone A and vice-versa. This may effect Zone A performance due to extra load. During disaster recovery situations every second counts and restoring cloud resources can become a matter of survival for a particular business. This requires an autonomous seamless and resilience carrier cloud broker-age solution as shown in Figure 2, where multiple clouds are connected to a single cloud brokerage solution which can provision resources to accommodate different IaaS requests. Cloud service providers can provide scalable resources to accommodate the requirements within minutes. The entire process is required to be initiated dynamically. In order to achieve this, we have proposed a real-time cloud brokerage that provides seamless, autonomous (self-service and self-managed) resource provisioning for a contingency cloud to replicate the destroyed IaaS during a natural disaster. The proposed cloud brokerage will automatically trigger the deployment of contingency cloud resources and redirect the user request to the alternative cloud services.

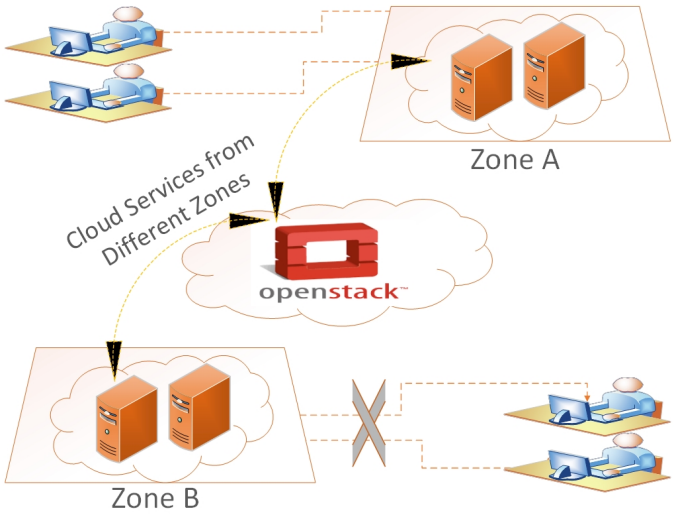

Figure 1. Cloud Service Scenario for Disaster Management

IV. PROPOSED ARCHITECTURE

There are multiple solutions of data recovery in cloud computing that depends on user requirements and IT budget or you can mix multiple approaches according to your unique scenario. As, we have merged two approaches, one is for rapid data recovery while other is for cloud controller lost. We have proposed a carrier cloud brokerage solution for federated cloud portability and to provide resilient infrastructure services on demand. It provides synchronization among multiple clouds platforms through a central broker. The proposed carrier cloud brokerage solution will on-the-fly identify and select the best available resources in the region and migrate the load to it. Multiple options for cloud services available in a particular zone are given to the users to select resources based on their preferences. If one server in a specific zone collapses, the user will automatically be connected to an-other server transparently.

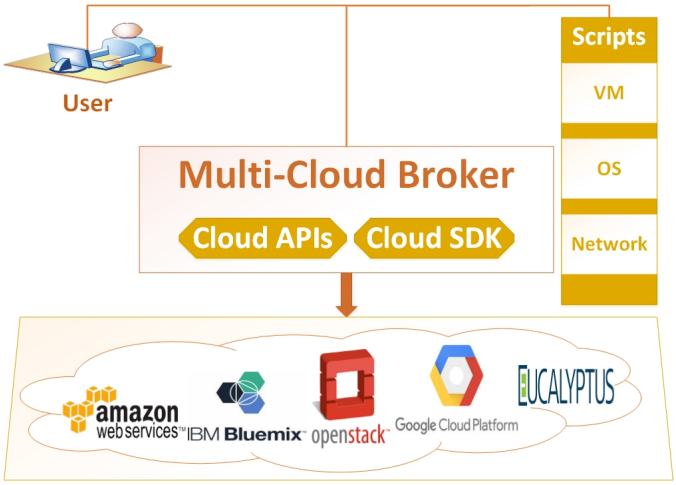

Figure 2. Muti-Cloud Brokerage Solution

We have developed a brokerage solution using libcloud SDK for on-the-fly resource provisioning of cloud services during disaster recovery. The libcloud SDK uses native cloud APIs thus is compatible with a variety of cloud platforms. Apache libcloud is an open-source project that is helpful for inter cloud resource management [12]. Table II explains different terminologies used in libcloud SDK [12]. The architecture of the proposed framework is given in Figure 3.

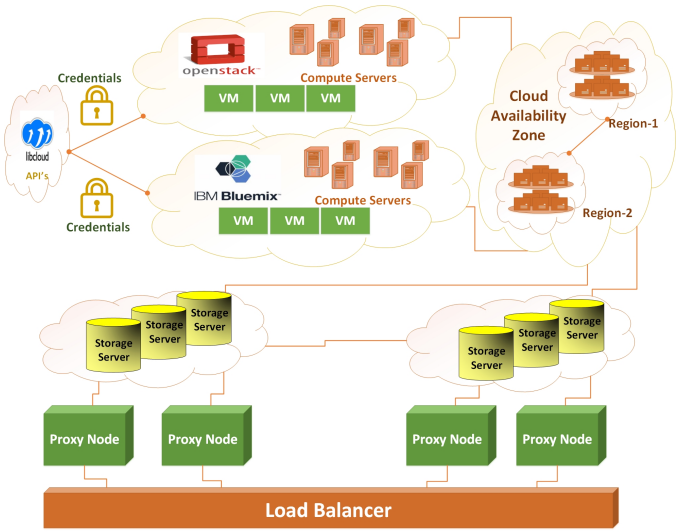

Figure 3. Carrier Cloud Architecture

V. Implementation

We have designed and implemented an autonomous and resilient cloud brokerage solution to provide on the fly IaaS resource provision during disaster management and recovery. We have implemented test-beds using open source cloud software. We have developed and implemented a cloud brokerage solution that can connect to multiple clouds. To implement the cloud testbed, controller, compute, network and storage nodes are used. Controller node is used to control all cloud services and activities. Compute nodes provide computing services as all VMs will run on compute servers. Network nodes will provide networking services to VMs. While storage nodes are used to store and retrieve data. We have used multiple compute and storage nodes to replicate seamless services in case of any failure.

We have assumed that each compute node is hosting virtual resources in a separate zone. This is in case one compute node in a particular zone fails or collapses as a result of natural disaster. The cloud brokerage will auto-initiate the recovery process through the cloud brokerage to identify and locate the suitable resource to migrate the business services. The storage services will also be replicated using the backup storage data to store user data on other nodes. In case of any storage server failure, users can still retrieve their data from other nodes.

VI. Performance Evaluation

In order to evaluate the performance of the proposed framework, a benchmarks approach is used to measure the performance. The benchmarks framework is the collective experiences of the cloud services to observe cloud capacities and response while accommodating unforeseen circumstances. As, every cloud has its own resource limitations and we cannot allocate more resources than its capacity. Therefore, benchmarking is the best practice to measure cloud performance for cloud users and provides an awareness for underloaded or overloaded exposure. It facilitates the process of planning, monitoring and evaluation of cloud services in an appropriate manner across the cloud users. The main objective of this benchmarking is to validate the cloud services, identify the gaps between cloud services and replicate the cloud services during disaster recovery.

Since we have assumed that each compute host is located in a separate zone, if any cloud compute host server is down in case of disaster, the requests will be automatically redirected to another available compute host server via dynamic VM consolidation. This scheduler service will automatically catch up the other active compute host on-the-fly. Similarly, cloud storage servers can also be replicated and replaced using an on-the-fly mechanism during the disaster recovery process. As, cloud computing platforms support different disaster recovery approaches. We have implemented the following two type of approaches that will support a DR plan in a cloud environment.

A. VM Migration for Compute Service

We have implemented nova service with multiple compute servers to support VM migration in the disaster environment. This nova service will provide compute service to the cloud to manage the instance life cycle and scheduling [22].

Table II
TERMINOLOGY IN LIBCLOUD

Terminology	Description
Node	Represents a virtual machine/ virtual server.
Node Image	Represents an operating system.
Node Size	Represents the hardware configuration of the virtual server, including CPU, RAM and Disk.
Node Location	Represents a physical location of virtual servers. In our case, VM availability zone is nova.
Node State	Represents a node state; either node is running, terminated or rebooting etc.

Multiple compute servers are used to facilitate the migration services from one region to another in case of disaster. When one compute server goes down, VM will migrate to another compute host server. This DR mechanism is fully transparent and scalable during a disaster.

We have performed the following tests to measure the performance of server migration in case of disaster management.

1) Performance measurement of boot and migrate server: In this experiment, we have deployed multiple compute servers to deal with the disaster situation and these are compute servers are available in nova compute zones. We have launched a server on an available compute node. When we detect any disaster then we will migrate this server to another available compute server in the availability zone. Once the VM is migrated to a new compute server, then this experiment will confirm the resize of the VM on new compute host and later delete the VM.

We have measured the performance of this scenario with five iterations with four atomic actions i.e. nova.boot_server, nova.migrate, nova.resize_confirm and nova.delete_server. As shown in Figure 4, this performance experiment results in a maximum time of 65.573 sec to complete this task, while 49.502 sec is the minimum time to complete this experiment.

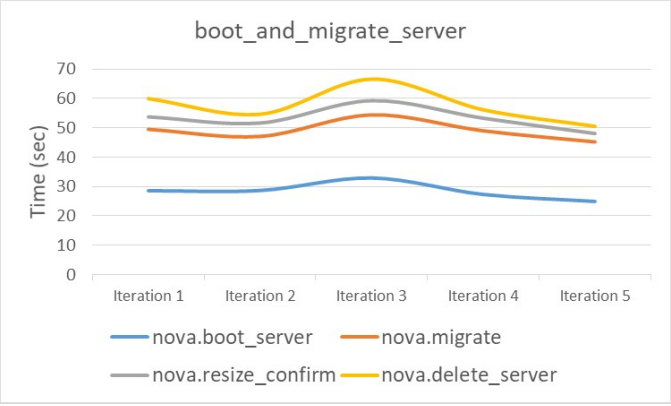

Figure 4. Boot and Migrate Server

B. Replicas for Storage Service

We have implemented swift services with a cloud to store unstructured data. This storage service is fully distributed and provides high availability and scalability to store and get data on demand.

Swift consists of proxy server and storage server [22]. Proxy server will forward the data to the storage server and the storage server will store that data. Ring will decide the optimal storage server to store data. For example, a user requests data storage on cloud swift. All swift replica storage servers will store the data. When users request that the data, an appropriate storage server will respond to the user with that data. A replicator is used to recover data in case of disaster recovery. This replicator detects the lost data from storage server and replicates this data to another storage server as a temporary measure. When the damaged server is replaced with a new server then the replicator replicates the data on a fresh server and removes the data from the temporary server. The following servers are also part of swift service.

- Object Server: is responsible for storage, retrieval and deletion of objects [22].
- Container Server: is responsible for listing of objects [22].
- Account Server: is responsible for listing of containers [22].

we have performed the following tests to measure the performance of this storage service in case of disaster management.

1) Performance measurement of storage container creation with objects and download objects: In this experiment, we have used multiple storage nodes to deal with disaster emergency problems. On these storage nodes, we have created a storage container to store unstructured data on multiple objects. These objects are responsible for storage, retrieval and deletion of data. After creation of these objects we have also measured the download time of the data.

We have measure the performance of the proposed solution with six iterations with these three atomic actions i.e. swift.create_container, swift.create_5_objects and

swift.download_5_objects. As, Figure 5 represents this performance experiment results with maximum time is 2.448 sec while 0.533 sec is minimum time to complete this experiment.

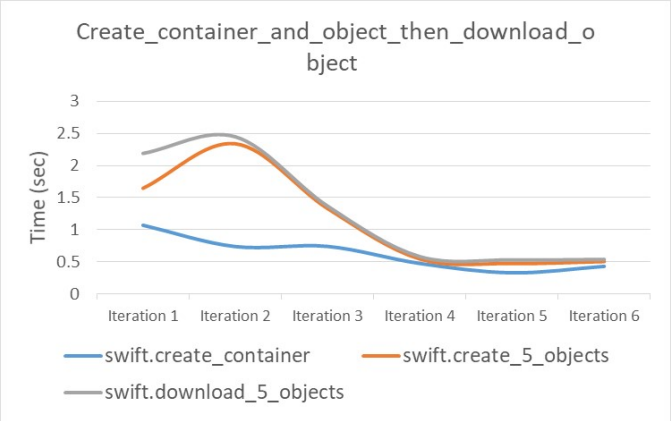

Figure 5. Container creation with objects and download objects

2) Performance measurement of storage container creation with objects and then delete all : In this experiment, we have measured the performance of the proposed solution as per container creation with specific number of objects and then delete both things (i.e. object and container) but make sure before deleting the container, you need to delete objects where data are stored. We have performed four iterations with the same atomic actions (i.e. swift.create_conatiner, swift.create_5_objects, swift.delete_5_objects and swift.delete_conatiner). As, Figure 6 represents this performance experiment resulting in a maximum time is 1.329 sec while 0.787 sec is minimum time to complete this experiment.

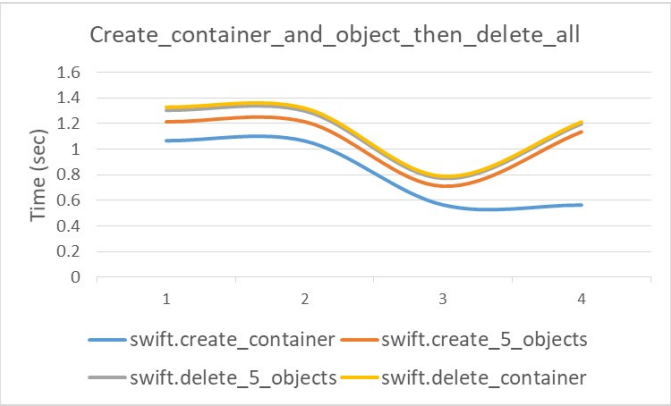

Figure 6. Container creation and deletion with objects

VII. Conclusion and Future Work

Business continuity is a major requirement for all organizations in case of disaster data losses during disasters can create a huge loss for the business including financial and reputation damage. Various organizations often invest certain amounts to validate data recovery quickly during disaster. However, today cloud computing offers flexible solutions to fulfill the business needs according to business requirement. In this paper, an autonomous seamless cloud solution is presented for data recovery during disaster management. Currently, this solution is implemented for a private cloud that fit for the single organization requirements. We have per-formed various tests to validate our solution which significantly provides resilient and robust cloud performance. In future work, we can move to PaaS with light-weight technology like container (docker) which is faster than traditional VM's.

Notes

Abbreviations

Business Continuity and Disaster Recovery (BCDR), Cloud Data Management Interface (CDMI), Disaster Recovery (DR), Recovery Time Objective (RTO), Recovery Point Objective (RPO), CSP (Cloud Service Provider), Infrastructure as a Service (IaaS), Open Cloud Computing Interface (OCCI), Open Virtualization Format (OVF), Topology and Orchestration Specification for Cloud Applications (TOSCA), Platform as a Service (PaaS), Software as a Service (SaaS), Elastic Block Store (EBS), Domain Name System (DNS), Content Delivery Network (CDN), Business Continuity (BC).

Conflict of Interest

The authors declare that they have no conflict of interest.

Authors' Information

Sonia Shahzadi received her BS and MS degrees from University of Gujrat, Pakistan, in 2013 and 2016 respectively. She is currently associated with Swan Mesh Networks Ltd, Research and Development, London, UK. Her research interests include Cloud Computing and Mobile Edge Computing.

Muddesar Iqbal is Senior Lecturer in Mobile Computing in the Division of Computer Science and Informatics, School of Engineering. He won an EPSRC Doctoral Training Award in 2007 and completed his PhD from Kingston University in 2010 with a dissertation titled Design, development, and implementation of a high-performance wireless mesh network for application in emergency and disaster recovery. He has been a principal investigator, co-investigator, project manager, coordinator and focal person of more than 10 internationally teamed research and development, capacity building and training projects. He is an established researcher and expert in the fields of: mobile cloud computing and open-based networking for applications in Education, disaster management and healthcare; community networks;

and smart cities. His research interests include 5G networking technologies, multimedia cloud computing, mobile edge computing, fog computing, Internet of Things, software-defined networking, network function virtualization, quality of experience, and cloud infrastructures and services.

George Ubakanma joined London South Bank University (LSBU: then South Bank Polytechnic) in 1992 as a Lecturer in Operating Systems and Networking. He is currently a Senior Lecturer, specializing in Database Management; Business Intelligence Architecture; Systems Analysis and Design. Current departmental duties also include the role of Course Director for several of the Postgraduate (MSc) courses in the Department of Informatics. As well as the course management and student facing responsibilities that the Course Director role brings, He also maintains an active cross-Faculty role as a Departmental and Faculty Academic Integrity Officer. He is currently Co-Chair for LSBU's Academic Integrity Co-ordinators(AIC), this role requires regular cross-Faculty/University contact, together with the research and dissemination of best practice with AIC's from all faculties as well as the University Registrar's Team. He has developed and co-authored various MSc courses for the Department of Informatics (previously: School of Computing & Mathematics). He has been course director for several MSc courses, as well as managing the BCS Examinations at LSBU. His course director's role has also required undertaking a commitment to conduct recruitment and marketing responsibilities overseas, particularly in India and South East Asia, as well as in Africa, particularly in Nigeria and Ghana. He is currently actively involved in the restructuring of the Informatics department's postgraduate and undergraduate course provision. He has also undertaken the role of internal examiner/reviewer for the validation of other LSBU courses. He is also an external examiner at other UK universities and colleges.

Tasos Dagiuklas is a leading researcher and expert in the fields of Internet and multimedia technologies for smart cities, ambient assisted living, healthcare and smart agriculture. He is the leader of the SuITE research group at the London South Bank University where he also acts as the Head of Division in Computer Science. Tasos Dagiuklas received the Engineering Degree from the University of Patras-Greece in 1989, the M.Sc. from the University of Manchester-UK in 1991 and the Ph.D. from the University of Essex-UK in 1995, all in Electrical Engineering. He has been a principle investigator, co-investigator, project and technical manager, coordinator and focal person of more than 20 internationally R&D and Capacity training projects with total funding of approximately $5.0m$ from different international organizations. His research interests include Smart Internet Technologies, Media Optimization across heterogeneous networks, QoE, Virtual Reality, Augmented Reality and cloud infrastructures and services.

REFERENCES

[1] T. Wood, E. Cecchet, K. K. Ramakrishnan, P. J. Shenoy, J. E. van der Merwe, and A. Venkataramani, "Disaster recovery as a cloud service: Economic benefits & deployment challenges.," *HotCloud*, vol. 10, pp. 8–15, 2010.

[2] P. Kokkinos, D. Kalogeras, A. Levin, and E. Varvarigos, "Survey: Live migration and disaster recovery over long-distance networks," *ACM Computing Surveys (CSUR)*, vol. 49, no. 2, p. 26, 2016.

[3] "Sendai Framework for Disaster Risk Reduction 20152030." http://www.unisdr.org/files/43291_sendaiframeworkfordrren.pdf. Accessed: 2017-11-02.

[4] "Cloud-Based Disaster Recovery Emerging as Top IT Priority, White Paper." https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/whitepaper/cloud/idg-dr-whitepaper-0515-final.pdf. Accessed: 2017-11-02.

[5] "Cloud Disaster Recovery: Public, Private or Hybrid Cloud Solutions Supporting Disaster Recovery, White Paper." https://www.datalink.com/getattachment/a00eea92-bde3-4033-92e1-7e31dfd8f504/Cloud-Disaster-Recovery.aspx. Accessed: 2017-11-02.

[6] "Best Practices in Healthcare IT Disaster Recovery Planning, White Paper." https://www.cleardata.com/wp-content/uploads/2014/08/Best-Practices-Healthcare-IT-Disaster-Recovery-Planning.pdf. Accessed: 2017-11-02.

[7] A. Prazeres and E. Lopes, "Disaster recovery–a project planning case study in portugal," *Procedia Technology*, vol. 9, pp. 795–805, 2013.

[8] F. Fowley, C. Pahl, and L. Zhang, "A comparison framework and review of service brokerage solutions for cloud architectures," in *International Conference on Service-Oriented Computing*, pp. 137–149, Springer, 2013.

[9] D. Petcu and A. V. Vasilakos, "Portability in clouds: approaches and research opportunities," *Scalable Computing: Practice and Experience*, vol. 15, no. 3, pp. 251–270, 2014.

[10] "Jclouds." http://jclouds.apache.org/. Accessed: 2017-11-02.

[11] "δ-Cloud." http://deltacloud.apache.org/. Accessed: 2017-11-02.

[12] "Libcloud." http://libcloud.apache.org/. Accessed: 2017-11-02.

[13] "Fog." http://fog.io/. Accessed: 2017-11-02.

[14] "Dasein." http://dasein-cloud.sourceforge.net/. Accessed: 2017-11-02.

[15] "Simple Cloud." http://addhrere:/. Accessed: 2017-11-02.

[16] F. Meireles and B. Malheiro, "Integrated management

of iaas resources," in *European Conference on Parallel Processing*, pp. 73–84, Springer, 2014.

[17] "Cloud abstraction API's." https://sites.google.com/site/jmathaiy/references-and-learning-guide/cloud-abstraction-apis. Accessed: 2017-11-02.

[18] "Multi-cloud." https://medium.com/@anthonypjshaw/multi-cloud-what-are-the-options-part-1-low-level-abstraction-libraries-ce500f29120f. Accessed: 2017-11-02.

[19] K. B. Nayar and V. Kumar, "Benefits of cloud computing in education during disaster," in *Proceedings of the International Conference on Transformations in Engineering Education*, pp. 191–201, Springer, 2015.

[20] O. H. Alhazmi, "A cloud-based adaptive disaster recovery optimization model," *Computer and Information Science*, vol. 9, no. 2, p. 58, 2016.

[21] R. Jaluka, D. Meliksetian, and M. Gupta, "Enterprise it as a service: Transforming the delivery model of it services," in *Cloud Computing in Emerging Markets (CCEM), 2016 IEEE International Conference on*, pp. 32–39, IEEE, 2016.

[22] "OpenStack." www.openstack.org/. Accessed: 2017-11-02.