2014 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)

IEEE CLOUD COMPUTING for Emerging Markets Conference

15 & 17 OCTOBER 2014 • BANGALORE, INDIA

IEEE Catalog Number: CFP14CCM-ART
ISBN: 978-1-4799-6141-2

IEEE CLOUD COMPUTING

IEEE
Advancing Technology for Humanity

2014 IEEE International
Conference on Cloud Computing in Emerging Markets (CCEM)

IEEE Catalog Number: CFP14CCM-ART
ISBN: 978-1-4799-6141-2

Printed copies of this publication are available from:

Curran Associates, Inc
57 Morehouse Lane
Red Hook, NY 12571 USA

Phone: (845) 758-0400
Fax: (845) 758-2633
E-mail: curran@proceedings.com

Produced by IEEE eXpress Conference Publishing

For information on producing a conference proceedings and receiving an estimate, contact conferencepublishing@ieee.org

http://www.ieee.org/conferencepublishing

Welcome Message

It gives us great pleasure to welcome you to the Third Edition of the IEEE International Conference on Cloud Computing for Emerging Markets (CCEM)!

This conference is now established as an annual event that takes place in October and is unique in addressing the issues of Cloud Computing in Emerging Markets. We have a comprehensive conference program this year that spans academic research, industry expertise, startup innovations, government insights, and student projects.

As we approach the end of 2014, Cloud Computing is no longer a "nice to have" but a "must have" for most companies and institutions around the world. In every industry cloud computing is being seen as an essential paradigm for business transformation and as the basis for new systems of insight and engagement driven by the convergence of mobile computing, social media, and big data analytics. Emerging markets are playing an active role in this cloud transformation – be it in adoption across industry, government, or academia, in producing trained and talented professionals, in research and innovation, in meeting unique needs of developing countries, and in producing numerous startup companies. Hence CCEM, with its focus on emerging markets, provides a highly relevant and important forum to the entire spectrum of technical professionals and students engaged in cloud computing in emerging markets.

This year's conference continues to have at its core a set of high quality peer-reviewed technical papers. Out of more than 80 submissions received this year, only 8 were selected for oral presentation, indicating the very high standards of this conference. A further 25 papers were chosen for poster presentation. We thank all the authors who submitted excellent papers to this conference. We are indebted to the technical program committee both for their rigor of evaluation and the valuable feedback they provided to the authors. This year was also special in the wide variety of institutions, particularly academic institutions around the world, from which papers were submitted and got selected.

In addition to the peer-reviewed technical papers, CCEM has always had strong industry participation. This year's conference stands out in this regard with invited keynote talks from global experts at Accenture, Amazon, Cisco, EMC, Google, HP, IBM, IEEE, Netapp, Nokia, Prompt, and SAP. We particularly thank S.V.Sankaran from SAP and Yatheendranath Tarikere from IEEE for their efforts in identifying and convincing many of these invited keynote speakers to present their valuable insights at CCEM.

CCEM continues to showcase the high level of innovation and energy among startups in emerging markets. We are grateful to Rohan Joshi from Wolken Software for his continued enthusiasm and efforts to drive participation of both prominent venture capital companies as well as exciting startup companies at CCEM. Rohan was ably joined by Ajit Yohannan and Kanumury Radhesh from IBM in organizing this year's startup showcase. This year's CCEM also includes a doctoral symposium for the first time. We are grateful to Dr. Sriram Kailasam for organizing this symposium. Sri Chandra continued to be an active force behind the conference, promoting all aspects of the conference with the broad technical community and championing

the student competition that is being introduced this year. As in previous years, we also have a panel session with a diverse set of experts in Cloud Computing.

We would like to express our deep gratitude to all the members of the organizing committee for their efforts over many months, culminating in this conference. We should particularly call out the leadership and commitment of Kathy Grise to CCEM again this year, the dedication of Sai Dattathrani in planning and tracking actions for many months as Program Manager for the conference, and to Dillian Waldron for working on many of the logistics of the conference such as the venue, registrations, and program book.

We would specially like to thank all the sponsoring organizations for their support, backing, and enthusiasm for CCEM.

We welcome you all again and urge you to take full advantage of the rich opportunities to participate, learn, and network at CCEM 2014!

Dr. Gopal Pingali, IBM and Dr. D. Janakiram, IIT Madras
Conference Chairs

Dr. T.S. Mohan, Pragyan Labs & Dr. Rajkumar Buyya, University of Melbourne
Technical Program Committee Chairs

Welcome Message

On behalf of the IEEE Cloud Computing Initiative and the IEEE Cloud Computing for Emerging Markets (CCEM) Organizing Committee, welcome to the third IEEE CCEM conference in Bangalore, India!

In the space of a few years, cloud computing has changed computing—impacting everything from individual consumers, small- and mid-sized businesses, enterprises, and service providers, to governments. Enabled by the cloud, big data analytics has further disrupted business by giving managers unprecedented insights. A "next platform of computing," building on cloud computing, big data, mobile, social networking, the Internet of Things, and cyber-physical systems, will compose an infrastructure encompassing billions of users, millions of applications, exabytes of data creation per day, and ultimately hundreds of billions of connected things.

Holding CCEM in India provides an outstanding forum for discussion of the unique challenges and opportunities around emerging markets as they pertain to cloud computing and the "next platform." I invite you to enjoy our program and its rich offerings.

We hope that you have an excellent conference!

Stephen L. Diamond
Chair, IEEE Cloud Computing Initiative
Chair, IEEE Cloud Computing Standards Committee

Organizing Committee

Conference Chairs

Dr. Gopal Pingali
IBM

Prof. D. Janakiram
IIT Madras

Technical Program Chair

Dr. T.S. Mohan
India

Dr. Raj Buyya
University of Melbourne, Australia

Startup Showcase Chair
Rohan Joshi, Wolken Software
Ajit Yohannan, IBM

Industry Sponsorship Chair
Sri Chandra, IEEE

Project Office
Sai Dattathrani, IBM, India

Publicity Chair
Kathy Grise, IEEE

Invited Talk Chairs
S.V. Sankaran, SAP, India
Yathi Tarikere, IEEE

Doctoral Symposium Chair
Dr. Sriram Kailasam, SSN College of Engineering and technology, Chennai, India

Advisory Committee
Steve Diamond, EMC, USA
Kathy Grise, IEEE, USA
Harish Mysore, IEEE, India

Technical Program Committee
Kumar Bhaskaran, IBM Global Research Labs, Shanghai, China
Kamal Bhatttacharya, IBM Research, Africa
Hong Cai, ZTE TX, China
Rajdeep Bhowmik, Cisco, USA
Harriett Cao, IBM Research, China
Renato Fontoura de Gusmao Cerqueira, IBM Research, Brazil
Rong Chang, IBM Research, USA
Brian de renzi, IBM Research, Africa
Rahul De, IIM, Bangalore
Manish Gupta, IBM, Delhi
Masum Hasan, Cisco, USA
Raghu Hudli, Objectorb Technologies, India
Venkata Jagana, IBM, USA
Mark Karol, IEEE, USA
Chid Kollengode, Nokia, India
J. Lakshmi, IISC, Bangalore
Jayaprakash Lalchandani, IIIT, Bangalore
Ulisses Mello, IBM Research, Brazil
Veena B. Mendiratta, Bell Labs, Alcatel-Lucent, USA
Christine Miyachi, Xerox, USA
San Murugesan, University of Western Sydney, Australia
Maitreya Natu, TCS, Pune
Marco A. S. Netto, IBM Research, Brazil
C.S.R. Prabhu, NIC, Delhi
Rushi Prafull Bhatt, Amazon, Bangalore
Jon Rokne, University of Calgary, Canada
Chunming Rong, University of Stavanger, Norway
Sudha Sadhasivam, PSG College of Technology, India
Srinivasan Sengamedu, Amazon, India
Yogesh Simmhan, Indian Institute of Science, Bangalore, India
Arun Somasundara, BIT Mesra, Ranchi, India
Osamuyi Stewart, IBM Research, Africa
Jay Taneja, IBM Research, Africa
Sudarshan Tsb, Amritha Vishwa Vidyapeetham, Bangalore, India
Akshat Verma, IBM Research, Delhi
Doug Zuckerman, IEEE, USA

Table of Contents

A Cloud-Based Collaborative Platform Supporting Serviced-Enhanced Products for Emerging Markets 1
Bholanathsingh Surajbali, Adrian Juan-Verdejo, Spiros Alexakis, Holger Bar, and Markus Bauer

A Secure Mutual Authentication Protocol for Cloud Computing Using Secret Sharing and Steganography 9
Mrudula Sarvabhatla, Giri Murugan, and Chandra Sekhar Vorugunti

Analyzing User Behavior Using KeyStroke Dynamics to Protect Cloud from Malicious Insiders 17
Mahesh Babu Bondada and Mary Saira Bhanu S.

Autonomic Characterization of Workloads Using Workload Fingerprinting 25
Rahul Khanna, Mrittika Ganguli, Ananth S. Narayan, Abhiram R., and Piyush Gupta

Cloudy with a Spot of Opportunity: Analysis of Spot-Priced VMs for Practical Job Scheduling 33
Vedsar Kushwaha and Yogesh Simmhan

Contracts in Cloud Computing 41
Irene Kafeza, Eleanna Kafeza, and Epameinondas Panas

SLA-aware Provisioning and Scheduling of Cloud Resources for Big Data Analytics 49
Mohammed Alrokayan, Amir Vahid Dastjerdi, and Rajkumar Buyya

A Cloud Computing Service Level Agreement Framework with Negotiation and Secure Monitoring 57
V. Binu and N. D. Gangadhar

A Hybrid Protocol to Secure the Cloud from Insider Threats 65
Sriram M., Vaibhav Patel, Harishma D, and Nachammai Lakshmanan

A Novel Bio-Inspired Load Balancing of Virtual Machines in Cloud Environment 70
Ashwin T. S., Shridhar G. Domanal, and Ram Mohana Reddy Guddeti

Achieving Energy Efficiency by Optimal Resource Utilization in Cloud Environment 74
Devwrat More, Sanket Mehta, Pooja Pathak, Lokesh Walase, and Jibi Abraham

An Enhanced Cloud Computing Model for Patient Record Management in South Africa 82
Richard Millham

Big Data Infrastructure for Aviation Data Analytics 87
Anandavel Murugan, Dinkar Mylaraswamy, Brian Xu, and Paul Dietrich

Client Requirement Modeling Using Resource Broker Architecture in Cloud Computing Environment 93
Chetan Awasthi and Priyesh Kanungo

Cloud Based Self Driving Cars 99
Naveen S. Yeshodara, Nikhitha Kishore, and Namratha S. Nagojappa

Cloud Partner Selection Algorithm for Dynamic Cloud Collaboration 106
Pramod Mane and Abhay Ratnaparkhi

Cloudifying Apps - A Study of Design and Architectural Considerations for Developing Cloud Enabled Applications with Case Study 110
Narsimha Reddy Challa and Venkatesh Nuthula

DBMLE: Distance-Based Multi-Level Elliptic Routing Protocol for Ad hoc Networks 117
Kevin Joy Dsouza and Sujatha M.

Experimental Comparison of Three Scheduling Algorithms for Energy Efficiency in Cloud Computing 122
Sudhir Goyal, Seema Bawa, and Bhupinder Singh

MultiPaaS - PaaS on Multiple Clouds 128
Shashank Sahni and Vasudeva Varma

Network Telemetry Anonymization for Cloud Based Security Analysis - Best Practices 134
Sashank Dara

Outsourcing Resource-Intensive Tasks from Mobile Apps to Clouds: Android and Aneka Integration 141
Tiago Justino and Rajkumar Buyya

Privacy Preserving Third Party Auditing in Multi Cloud Storage Environment 149
M.S. Shashidhara and Jaini C. P.

Secure Data Mining in Cloud using Homomorphic Encryption 155
Deepti Mittal, Damandeep Kaur, and Ashish Aggarwal

Towards Realizing the Secured Multilateral Co-operative Computing Architectural Framework 161
Manu A. R., V K Agrawal, K N Balasubramanya Murthy, and Manoj Kumar M.

Author Index 175

A Cloud-Based Collaborative Platform Supporting Serviced-Enhanced Products for Emerging Markets

Bholanathsingh Surajbali, Adrian Juan-Verdejo, Spiros Alexakis, Holger Bar, Markus Bauer
CAS Software AG, Karlsruhe, Germany
{b.surajbali, adrian.juan, spiros.alexakis, holger.baer, markus.bauer}@cas.de

***Abstract*— Collaboration through sharing competencies and resources has been a key approach to both creating new competitive environments, as well as achieving the needed agility to rapidly answer to emerging market demands. Establishing proper collaborative networks for service-enhanced products is challenging given the wide diversity of business operations and involved resources. The creation of software solutions to support such collaboration is an effort-intensive task, especially when these solutions are designed to run in the cloud and to be highly customisable to different end-user scenarios. In this paper we describe how a cloud-based platform can support the creation of software solutions for the collaborative development and operation of highly customised and service-enhanced products. The platform has been designed based on numerous key requirements that have been generated from the analysis of different business scenarios from various use cases and domains, as well as general key requirements a cloud-based platform should provide to support collaboration among organisations.**

***Index Terms*— Collaborative networks, Cloud Computing**

I. INTRODUCTION

There is a growing trend in manufacturing to move towards highly customised products, even one-of-a-kind, which is reflected in the term *mass customisation*. Mass customisation refers to a customer co-design process of products and services which meet the needs/choices of each individual customer with regard to the variety of different product features. The development of these products typically requires a variety of competencies and resources, hardly available in a single enterprise, which calls for collaboration among several companies and individuals. Collaboration through sharing competencies, resources and services has been a key approach to both creating new competitive environments, as well as achieving the needed agility to rapidly answer to emerging market demands. Being able to rapidly react to a collaboration opportunity in fast changing market conditions is a key requirement for dynamic virtual organisations (VOs [1] also known as a network).

The notion of *glocal* enterprise emerged to represent the idea of thinking and acting globally, while being aware and responding adequately to local specificities, namely in collaboration with local stakeholders and customers for VO .

Software that runs in the cloud has gained significant attraction in the past few years representing the emerging paradigm for distributed value added services. With cloud computing, enterprises are given new opportunities to push virtual collaborative alliances to the next level, breaking down some barriers and enabling dynamic continuous collaboration that generates globally composed business value in terms of products and services. In this paper, we propose a collaborative cloud based platform, developed in the GloNet project [5, 25] and providing support for the mass customisation of service-enhanced products. The rest of the paper is organised as follows: in Section II describes the main functionality required for collaborative networks. Then in Section III we describe the essential design principles for a cloud-based approach for collaborative networks. Next, in Section IV we describe the main architecture of the GloNet platform, followed by Section V describing how external services can be integrated with the GloNet platform. Next, in Section VI we describe the GloNet platform implementation followed by a discussion of our approach and related work in Section VII. Finally we offer our conclusion in Section VIII.

II. MOTIVATION

To motivate the need of collaborative network (CN) in emerging market we use the GloNet EU project Charanka Solar Park plant use case scenario [6]. A Solar Plant represent a product that consist of a range of stakeholders from customers, project developers, construction and commissioning firms, equipment suppliers, service provision companies and many others, are considered to be either a member of a long term alliance of stakeholders involved in the area of the product. The core functionalities typically provided in the Solar Plant include: the amount of solar radiation to energy efficiency; location of plant through monitoring the system performance and device failures, while issuing warnings, e.g. email and/or text message notifications to the plant engineering/maintenance office and staff once failure happens. Solar Plant is an example of a complex product having a number of different stakeholders from the Virtual Breeding Environment (VBE [1] also known as a cluster) involved, varying from the equipment manufacturers, to the suppliers, project developers, construction and commissioning firms, service provision companies and many others who are experts in the different parts of the Solar Plant. The complexity of information/knowledge sharing among different VBE members of a Solar Plant requires effective collaboration and sharing of the information about complex products each contributing to specific goals within the VO.

In setting up the collaboration environment among VOs to provide service-enhanced product, it is important to provide for the following functionalities to:

1. **Set up a collaborative space for project among different stakeholders**. The Solar Plant need to be able to select VBE members and create collaboration on a product-enhanced service. To support such a collaboration,

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

stakeholders must be able to share knowledge (in forms of documents, specification, email) and competencies among themselves and new VBE members. Furthermore, stakeholders must be able to create more than one collaboration with other VBE members, based on the intended goal, and close the collaboration in case it is no longer required over time.

2. **Support different roles in a CN and permissions for accessing information and knowledge resources.** It is important for the Solar Plant owner to ensure all information shared in the VOs are kept confidential as some of the VBE members could be competitors on other goals. Appropriate mechanisms need to be in place through the identification of the property rights per resource and provision of authorised access for network members, according to their roles.
3. **Share knowledge and data among VOs.** VO members within the Solar Plant will require sharing information on product or service (for example standardised product definitions and processes), software tools, and lessons learned. As such, the system must support organisations in the process of sharing their resources, allowing that each organisation inside the VO can search, retrieve and update information and knowledge about e.g. templates, standard processes, ontologies, etc. The system should also include mechanisms to implement incentive policies to increase resources sharing.
4. **Support groupware-related tasks including appointments and tasks management.** Solar Plant collaborating members must be able to perform groupware related task. This can be reflected, in terms of sharing appointments, tasks to meet the project goals within a VO.
5. **Ensure data consistency between the VO collaboration space and the VO existing information system.** On creating a VO collaboration space, it should be noted the products and services are only shared across the VO members and VOs own system. It is important to ensure, consistent synchronisation of information across the VO collaboration space and the VO information system. For instance, in the use case scenario as a bidding document is changed across the VO collaboration space, the document must be updated across the VO information system, to ensure data consistency.
6. **Maintain knowledge of the VO during its life-cycle.** VOs must support and promote the collaborative design of products and services as well as the emergence of innovative solutions. For instance VOs must be able to define workflows and information maintained throughout the life cycle of the VO.
7. **Ease of Access to VO collaborative network.** VOs within the Solar Plant can be involved in more than one collaboration. As a result VO members should be able to use and switch between the collaborative network easily, without having to re-login every time they shift from one collaboration space to another.
8. **Supply interoperability.** Each VBE member of the Solar Plant system is likely to have different information system. Hence, it is important that their information system, ERP, CRM system are able to seemingly interoperate with the collaboration network platform; e.g. VBE members may exchange offers of their products and services over time, and such offers must easily be exchanged with the collaborative network.

III. A Cloud-Based Approach for Collaborative Networks

Cloud computing can provide a centralised platform [8], which can be accessed from anywhere, by any (authorised) collaborator, and at any time over the Internet. Cloud solutions are typically presented as exhibiting a number of advantages for emerging markets stakeholders in terms of:

- **Reduced initial costs.** Cloud computing provides cost reduction that particularly SME organisations can benefit from whereby they do not own the resources (server, storage) that produce the computing capabilities they require and pay for.
- **Ubiquitous, simple access through Internet.** With cloud computing, organisations are given new opportunities to push virtual collaborative alliances to the next level, breaking down geographical barriers and enabling dynamic continuous collaboration that generates globally composed business value.
- **Centralised platform for collaborative scenarios, integration hub.** Cloud computing provides a centralised platform, which can be accessed from anywhere, by any VO, and at any time over the internet.
- **Flexible cost models.** In addition, customers benefit from the fact that cloud computing employs a *pay-per-use* model. Customers do not have to do a high initial investment in hardware and software, instead they are typically paying monthly or yearly fees depending on the number of user licenses they need and/or the resources they consume.

It is essential for a cloud–based software solution to be designed considering the fundamental requirements of a collaborative workspace environment. As such we believe the following five key design principles [7] are essential in building a collaborative cloud environment support platform:

1. **Modularity and extensibility.** In cloud-based software solutions, software sub-systems are typically consolidated having a code-base that works for all customers. However in collaborative networks, this approach is not feasible, since customer specific configurations, data models, business rules and workflows, need to be tailored to each individual user need. As such, software solutions need to support programmatic extensible and customisable

features, namely allowing the addition and removal of modules for specific needs.

2. **Multi-tenancy.** Multi-tenancy is the capability of a software system to serve multiple customers or tenants (which in turn comprises multiple users). In essence, having a cloud-based approach, two potentially conflicting requirements need to be addressed. That is on one hand the cloud-based approach needs to leverage the economy of scale principle by employing a consolidated architecture that handles all customers uniformly, and on the other hand customers demand that the software they use can be tailored to meet their specific requirements and match with their highly-individual business and the processes they work with. This implies that both data and customisations have to be *isolated* on a tenant-based level by the cloud-based platform.
3. **Scalability and Availability.** For any collaborative environment, coming up with a scalable software architecture is a major concern. For cloud-based systems, this is even more critical. The software will be used by thousands of users in parallel, namely the expected average number of concurrent users per customer, multiplied by the number of customers having licensed the software. There are different ways to achieve scalability. One way is to *scale-up* the system, that is to move the software to more powerful servers (for example having more processing power, more RAM, more and faster storage) when the need arises. For a cloud software approach, a *scale-out* strategy is much more suitable. This means that the workload of the system can be distributed among several servers. A scaling-out architecture offers two main advantages. First, standard and cheap off-the-shelf servers can be used instead of expensive high-end servers. Second, it offers higher availability, since many instances of the same application server are deployed.
4. **Security and Trust.** An important issue in adopting a cloud approach is ensuring data security. For collaborative solutions security mechanisms need be designed in a way that they still allow for data sharing among users that want to collaborate.
5. **Network access.** Cloud architectures need to be designed for network access from the ground up. This concerns the end-user interfaces (web-based UIs) as well as programmatic access (web services).

IV. The GloNet platform

The GloNet platform [25] is built according to functionality required by collaborative networks as discussed in Section III and the five key principles a cloud-based approach should take into consideration. Fig. 1 illustrates the main GloNet collaboration space and its sub-modules - single-sign-on, user permissions, synchronisation, service-enhanced-product and process workflow modules as well as the integration module – proxy-based and mash-up integration.

It uses a framework-based approach: it defines an application architecture and provides a number of components that implement the basic building blocks of this architecture. In addition, the platform is extensible, providing customisation mechanisms as well as extension and integration mechanisms for additional modules and services that may complement the basic features of the platform.

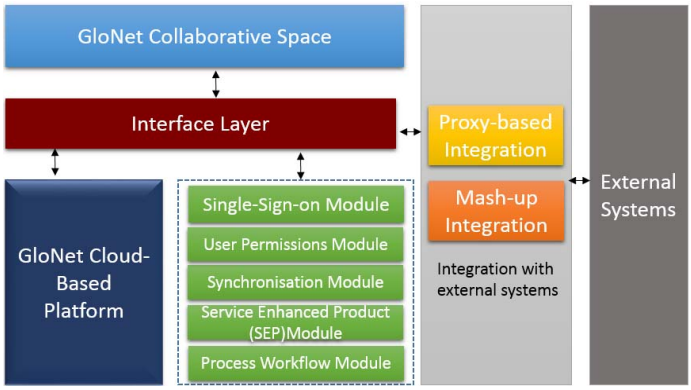


Fig. 1. GloNet system overview

A. GloNet Platform Architecture

The GloNet platform follows a layered architecture [2], whereby a system is decomposed into several distinct layers or *tiers* that can be developed, maintained and (often) deployed independently from each other - the presentation layer, the application layer and data layer. The *presentation layer* focuses on interacting with the user through a graphical user interface. It displays data and collects user input and commands. The application layer or *business logic layer* provides operations that implement the processes and operations that the software solution provides. The *data layer* encapsulates the storage and provides access to the persistent data of the solution. In most cases this layer makes use of a database management system.

The most important modules of the GloNet cloud-based platform are the following:

- The *data access* module that encapsulates database specifics. It enforces tenant isolation and implements a permission system, thereby strictly controlling the access to the data stored in the databases. As a clean interface, it provides a database and platform neutral simplified SQL-like language for data access including a number of helper functions to simplify the access to more complex data structures.
- The *server core* module, which builds the backbone of the backend implementation of the GloNet platform. It provides mechanisms to register and unregister additional modules, which in turn may provide additional operations. It is also responsible for enforcing a strict security policy by providing and verifying security contexts (for example, based on user and tenant credentials) when executing any kind of operation. Furthermore, it makes use of the data access module to manage generic (extensible) data objects.
- A number of *business operation* modules provide implementations for the basic services of the GloNet platform. Such operations include infrastructure-related operations (user and tenant management, permission management), but also operations concerning more end-

user related features (e.g. document management, calendar management). New modules can be registered using mechanisms provided by the server core module, creating an extensible platform.

- Any operation from the business operation modules or the server core with a publically defined interface is exposed via a number of *external interfaces*, namely in-process method calls, RMI and SOAP-style web services. A subset of those is also accessible as REST-style web services. This way, the GloNet platform provides a service-oriented *application programming interface (API)* suitable for a large variety of use cases and implementation technologies.
- The presentation layer of the GloNet platform provides a framework for creating and presenting a web-based user interface. The *presentation logic* module provides generic user interface functionality like reusable controls, the application frame, and configurable list and form views.

The platform supports customisation facilities on all layers of the software system:

- *User interface*. The GloNet platform allows users to define and customise their user interface to match with their corporate branding, that is adapt rather simple things like logos, colours and fonts, or to better reflect their internal nomenclature. Users can also modify the layout and ordering of the presented information in order to put an emphasis on business critical information.
- *Business logic*. The platform allows users to implement their daily business needs, by supporting customisation of business logic layer rules. For instance, a VO member in could define rules that are used to evaluate the rating or potential of a customer. In such situation, one VBE member's rating rule might be based on the yearly turnover with that customer, whereas another VBE member might prefer to make that rating dependent on the number of sold licenses for a software company or the overall value of profitable insurance contracts for an insurance company.
- *Data model extensions*. The platform allows VBE member users to store additional information in the system. In these cases, the platform supports extensions of the data model using custom objects or custom fields for each user.

B. GloNet Collaboration Space System

The GloNet collaboration space (as shown in Fig. 2) follows a layered architecture consisting of the presentation layer, business layer and data layer, in line with the GloNet platform architecture.

Presentation Layer. The *presentation layer* focuses on interacting with the user through a graphical user interface. It displays data and collects user input and commands.

Business Layer. The *business logic layer* provides collaboration operations that implement the processes and operations that the software solution provides. The EIMInterface [5] in GloNet is the main interface to communicate with the server. In order to support the collaboration space, the EIMInterface has been enhanced to support a context mechanism. As such whenever a business operation is invoked the latter is checked if it is a collaboration space or non-collaboration space business operation. In case it is of collaboration space business operation, using the right plug-in the user permission is checked and the associated data is then synchronized between the collaboration space and the GloNet platform or external information system.

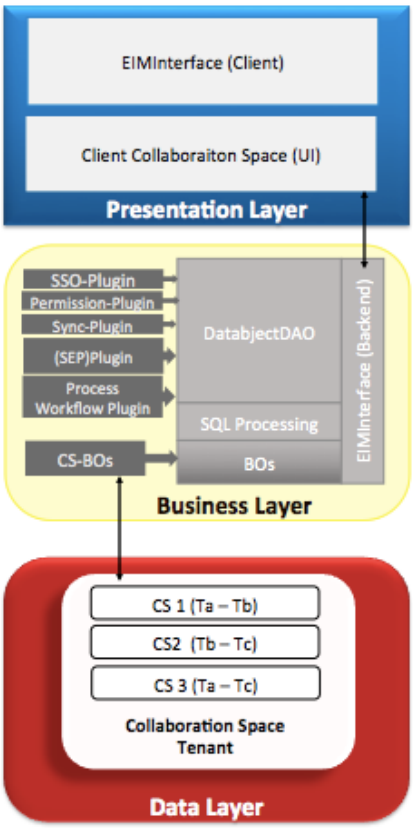


Fig. 2. GloNet Collaboration Space System

Furthermore it should be noted the context mechanism restricts data exposed to the user by providing access only to the data contained in his/her collaboration space. The context mechanism uses a number of key-value pairs bound to the EIMInterface which can contain for example, a collaboration space identifier and value represents the name identifier of the collaboration space. On the client side, the EIMInterface provides context through the EIMInterface that is add context meta-information in a key-value pair structure while invoking the server EIMInterface. The business layer contains three main components, the Data-Object-DAO, SQL processing and business operations (BOs). To support data object processing transparently, each of the components have corresponding plugin. The Data-Object-DAO component processes meta-information received through the EIMInterface using five plugins:

- *SSO-plugin* connects to the SSO module which provides Single-Sign-on functionality to enable users to connect to

collaboration space environment using one login mechanism (see Section C);

- *Permission-plugin* connects to the Permissions module which provides extensions to the server's native access rights management for the user in the collaboration space (see Section D);
- *Sync-Plugin* connects to the Synchronisation module which ensures data are kept synchronised in multiple tenants. (see Section E);
- *SEP-plugin* connects to the SEP module is responsible to associate SEP module to the collaboration space (see Section F); and
- *Process Workflow-plugin* connects to the Process Workflow module which maintains all product business and process information; (see section G)

Data Layer. The *data layer* maintains the collaboration space shared information. The CS uses a single Tenant to hold all the CS. This design ensures low administrative overhead in creating and deleting CS.

C. Single-Sign-On Module

The single-sign-on (SSO) mechanism allows users from the collaboration space to authenticate only once, thus making collaboration transparent allowing collaborative access decisions based on the user's identity. To provide this mechanism, the *OpenID* mechanism [4] for decentralised authentication and identity management is used. Using OpenID, the platform is able to provide SSO service integration, which enhances the usability of applications built on top of the platform.

D. GloNet Users Roles and Permissions

The GloNet platform provides a framework for the authentication and authorisation of its users. These are classified in three main categories:

- *Users, groups and resources*. These are called permission owners. Users can authenticate themselves with the platform's authentication methods. Users can be organised into groups. For each user, the platform maintains some basic attributes.
- *Security contexts*. After authenticating a user login, the platform creates a security context. Any operation runs within this security context. The platform provides different classes of security contexts, with user contexts always associated with an authenticated user and a user can only perform privileged operations based on the privileged contexts set to the user.
- *Permissions*. Most data elements and operations are associated with permissions. Permissions are used for authorising the access to data or the invocation of operations within the platform. For example, for permissions related to financial details, such as an employee's salary, the platform ensures only people from the HR department can be granted the permission to change the salary; however the platform ensures administration members are able to read the value in order to give out the pay checks.

E. Synchronisation Module

The collaboration platform uses a synchronisation module to allow synchronising data between the GloNet platform/VOs external databases and the collaboration space (as shown in Fig. 3). With data we mean individual records as well as relations between those records.

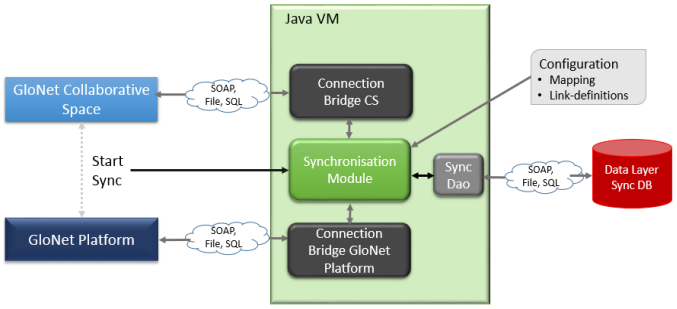


Fig. 3. Synchronisation module

As the software tracks the state of records and their relations it works highly efficient, it only transfers data that has really changed. At the beginning of a synchronisation session the module obtains the state from each of the two synchronised systems and compares them with the known state of the last session. The ConnectionBridge module controls the communication between the GloNet platform and the collaboration space system to be synchronised.

F. Service-Enhanced Products (SEP) Module

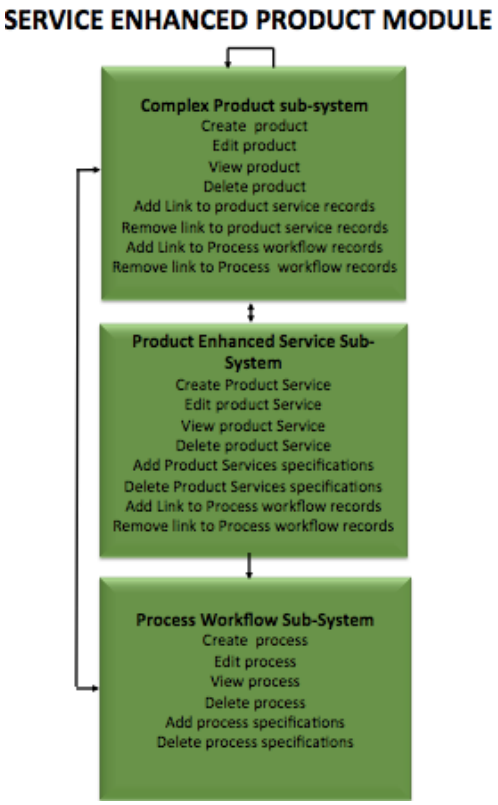


Fig. 4. Service enhanced Product (SEP) module

The service-enhanced products (SEP) module as shown in Fig. 4 provides the Product Life Cycle Management of complex

products (e.g. solar power plant and intelligent buildings) in terms of description associated sub-products, processes and services from the actual concept through events generated through maintenance, associated tasks as well as operational data, collected over its conceptualisation. Moreover, VOs have the ability to capture, analyse, and react to service data from early designs of a product by having data, information and related knowledge on service, events such as repairs and optionally warranty of products for stakeholders.

G. Process Workflow Module

The process workflow module maintains all product related processes, e.g. lessons learned to perform certain physical process or to take certain decision in the environment are maintained. It stores all generated knowledge on each SEP for the VOs as a business process workflow in terms of: i) *business services* –information about the business services related to product with the sequence of steps to accomplish a specified task; ii) *generated knowledge*, in terms of SEP information (set of processes, e.g. for ordering, purchasing, maintenance) and Service-Enhanced-Product-based VBE network information containing information related to the VBEs (e.g. on registering members to create a new VO).

V. Mechanisms for Integrating Existing Services in GloNet

The GloNet platform provides two key approaches to integrate external components or systems: *proxy-based* and *mashup-based*.

A. Proxy-based Mechanism

In the proxy-based mechanism each external service has a proxy within the GloNet platform that performs the mapping between the GloNet platform and the external service as shown in Fig. 5. The proxy-based approach works in the following way:

- An event in the GloNet platform triggers the invocation of an external service. For example, such an event can be the execution of a command in the GloNet user interface or a condition (business rule) in the GloNet data model.
- The GloNet platform consults a service registry to identify the external service to be called and associates a suitable service proxy plugin with it.
- The service plugin collects and assembles the data needed as input for the external service. Usually it also transforms the data into the form required by the external services (data mediation).
- It then invokes the services using a suitable web service protocol (SOAP, REST). The actual invocation can either be supported by generic invocation components provided by the GloNet platform, or it can be executed by the plugin itself. The latter also allows for complex invocation protocols.
- The external service executes. Upon termination, the service plugin receives the output of the service execution, transforms it back into a form suitable for the GloNet platform and its components.
- The results of the service invocation can then be passed to the GloNet user interface or used to update the GloNet data model.

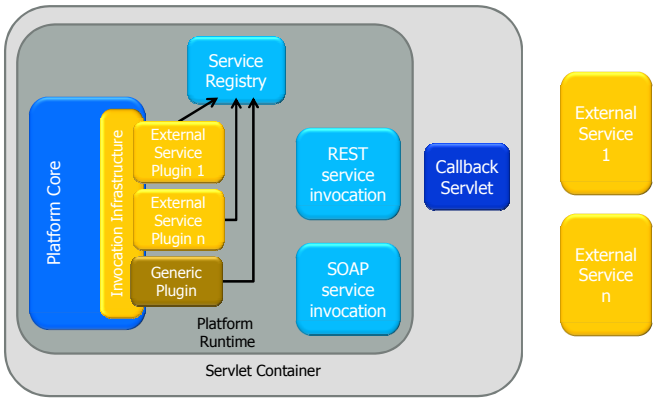


Fig. 5 External service integration

The invocation of an external service can be synchronous or asynchronous. While services that provide data or computations which should be displayed within an application are typically invoked synchronously, there might also be external services that run for a long time (minutes to days in case that the service triggers workflows that involve human interaction). The latter type of services are executed asynchronously. After the long-running service has been executed, a special callback servlet within the GloNet platform must be notified by the external service. This notification contains a transaction ID that identifies the context of original invocation (e.g. the user that triggered the invocation) as well as a one-time key that can be used by the external service in order to authenticate at the GloNet platform's security system.

For specific scenarios, generic plugins can be implemented. These plugins will then make a number of assumptions on the external services, i.e. depending on the scenario a service will have to implement a specific interface in order to be integrated. A good example might be services that provide sensor or profiling data for pieces of equipment deployed in a GloNet product. The generic plugins can be configured in a declarative way, which allows an integration of the external service without extending the GloNet platform's code base with the service-specific plugin implementation.

B. Mash-Based Mechanism

For the mash-based approach, external services make use of their own user interface that can be plugged-in to the GloNet platform. To exchange data with the GloNet platform, the provided web service API can be used. In order to interact with the GloNet application, basic user interface context information is exposed using a RESTful interface [3]. The mashup-based external service integration works the following way (see Fig. 6):

- An embedded browser widget (typically implemented using an IFrame) is added to the form. This browser widget

displays the user interface of the external service. This widget together with metadata that is needed to configure some basic integration properties (e.g. the external service's base URL) are part of the declarative form specification

- When rendering the form, the presentation layer sets the external service URL (communication channel 1) together with some basic context information (see below). This leads to a browser-side request to the external service (communication channel 2). Context information is passed using HTTP GET parameters.
- The external service can query user interface context information for the user's session, like the currently open object data object or the current selection of a known table (communication channel 3).
- Triggered by user interaction with the external service UI, the external service can interact with the webservice interfaces (REST, SOAP) of the GloNet platform (communication channel 4), as well as querying the current context information (communication channel 3).

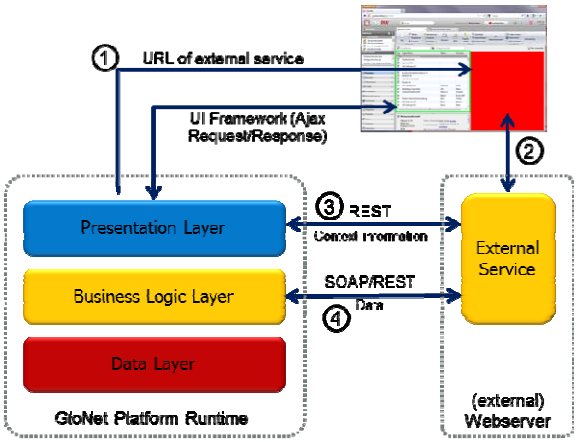


Fig. 6 Mashup-based external service integration

VI. GloNet Platform Implementation

The GloNet platform implementation employs a number of industry standards and off-the-shelf components which form the technical infrastructure of the GloNet platform. Fig. 7 illustrates the basic infrastructure for the deployment of GloNet. It consists of a *farm* of standard *Linux*-based PC-Servers that operate either as *Java* application servers based on *Apache Tomcat* or as database servers using *MySQL*. The presentation layer is currently implemented using server-side frameworks for AJAX-style user interfaces [4], Vaadin and Eclipse RAP. These components are standard building blocks of a Java based technology stack. Therefore, an instance of the GloNet platform can be deployed on almost any cloud infrastructure provider offering that allows for standard VM images as units of deployment. Adaptations to specific cloud infrastructure provider offerings are only necessary regarding the load balancing and VM administration interface. Alternatively the GloNet platform can be hosted in almost any computing centre that has some capability of operating Java-based application servers.

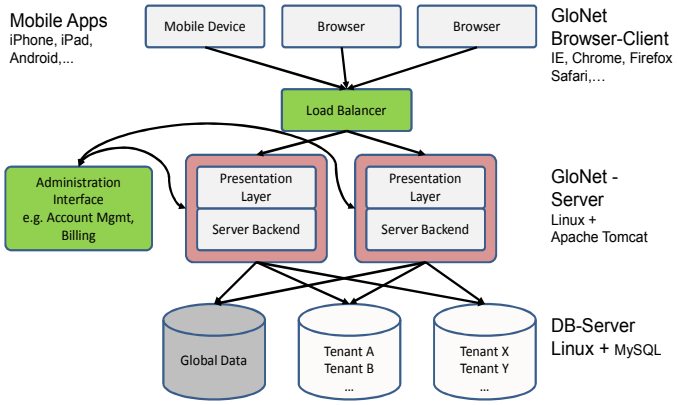


Fig. 7 GloNet platform deployment architecture

VII. Discussion and Related Work

Our proposed collaboration platform follows a top-down research approach and build on existing results from completed and on-going research projects for a best practice in Virtual Enterprise management for complex industrial units, supply-chain management, knowledge sharing, tracking complex processes and products as well as privacy issues. Our approach takes advantage of Cloud computing to provide a decentralised platform [22], which can be accessed from anywhere, by any (authorised) VO member, and at any time over the Internet. For SME this represents ease of use and availability of resources. Furthermore, using the integration tools [22], that is the mash-up and proxy integration modules, the collaboration space allows VOs to integrate their existing data (existing platforms, documents, know-how, specifications, tasks as well product and services history information). With the use of the synchronisation module, the data between the collaboration space and VO, and the collaboration space can assign rights and permission to each VO member while a VO member can be part of multiple collaboration spaces based on their goals. Furthermore, the collaboration is flexible allowing new VO members to join in and leave over time.

There are a number of research work in implementing cloud computing in manufacturing [23, 24] to enable the creation of integrated manufacturing networks. Despite the research and development work done in the last decade, the collaboration platform problem remains a practically unresolved issue, each addressing separate concerns of the collaboration domain. A number of research projects i-SURF [9], CONVERGE [10], COIN [11], SUDDEN [12] focussed on the support of collaboration across the supply chain to enhance planning, decision making and coordinate supply networks. Other research works such as ATHENA [13], SODIUM [14], COMMIUS [15], DIP [16] focused into development of web-based tools to enable interoperability of tools and applications. In addition to the web-based tools, ECOSPACE [17] offers service-oriented tools to manage complex daily work tasks in collaboration spaces. Significant research works such as LABORANOVA [18], SYNERGY [19], COIN [11], ECOLNET [20] have also looked into

creating tools to enable collaboration in teams, companies, organisations, networks and social communities in general. However, none of these approaches provide for resource sharing collaboration for VBE members and allowing them to collaborate on service enhanced products.

VIII. Conclusions

In this paper we have described the required functionality for a collaborative networks environment and proposed the GloNet platform, a cloud-based framework for creating applications for the collaborative design and operation of complex service-enhanced products. The platform is built according to a tailored cloud-based principle, taking into consideration specific requirements of collaborative networks. Additionally, the platform supports a VO collaboration for service enhanced product and is built according to a tailored cloud-based principle and provides mechanisms for the integration of external systems and services. Furthermore, our collaborative space platform offers a secure infrastructure for shared collaboration, allowing VOs to maintain information/knowledge in their existing systems (this includes the GloNet platform and other information systems) using the synchronisation module and is able to interoperate with existing VO platforms using the integration modules. The GloNet collaboration space represents a clear innovation for SMEs' to collaborate on enhanced product and services as well as management of VOs through better knowledge of product life cycle in a cost-effective manner.

Acknowledgment

This research work has been supported by the European FP7 Marie Curie ITN "RELATE" (Grant Agreement No. 264840) and GloNet project (FP 7 programme) Project No 285273, 2011-2014. The authors would also like to thank the contributions of the different partners of the GloNet project.

References

[1] Camarinha-Matos, L.M., Afsarmanesh, H., Ollus-Editors, M., Methods and Tools for Collaborative Networked Organisations, ISBN: 978-0-387-79423-5, Springer, 2008.

[2] Buschmann, F., Meunier, R., Rohnert H., Sommerlad, P. and Stal, M. 1996. *Pattern-Oriented Software Architecture*. A System of Patterns. Volume 1, Wiley.

[3] Hoyer, V. Stanoevska-Slabeva, K., Janner, T., Schroth, C. *Enterprise Mashups: Design Principles towards the Long Tail of User Needs.* In Proceedings of the IEEE International Conference on Services Computing. SCC 2008.

[4] Lange, F. *Eclipse Rich Ajax Platform: Bringing Rich Client to the Web*, Apress, 2008.

[5] GloNet project, https://sites.google.com/site/glonetproject/home-1

[6] GloNet Deliverable D1.2, "Specification of business scenarios" May 2012.

[7] GloNet Deliverable D3.1 "GloNet Platform Design Specification", June 2012.

[8] Cloud Standards Customer Council, "Cloud Computing Use Cases Version 1.0", http://www.cloudstandardscustomercouncil.org/use-cases/CloudComputingUseCases.pdf, White Paper 2011.

[9] Vieira, H., Goncalves, R-J., VALTE - Methodology for VALidation and TEsting of Supply Chain Software Components. Proceedings of the 15th International Conference on Concurrent Enterprising, ICE 2009.

[10] Rasoulifar, R., Eckert, C., Zolghadri., C. The need for a tool to exchange information in non-hierarchical network of the electronic industry: an European project. In Proceedings Design 2010, 17-20 May 2010, Dubrovnik, Catvat, Croatia.

[11] Kiauleikis, V., Romeika, G., Morkevi ius, N., Kiauleikis, M., Collaboration and Interoperability Services vs. Traditional Communication Means. In Journal of Science and Processes of Education. Collection of periodic and reviewed scientific articles issued Klaipeda Business College, Lithuania. 2011.

[12] Weichhart, G., Stary, C., Oppl, S., Modelling of Complex Supply Networks. In Proceedings of WETICE 06, pp. 265-268, IEEE Press, 2006.

[13] Goldstein, M.K. et al.: Implementing clinical practice guidelines while taking account of changing evidence: ATHENA DSS, an easily modifiable decision-support system for managing hypertension in primary care. Proc AMIA Symp. 2000.

[14] Pautasso, C., Alonso, G., Flexible Binding for Reusable Composition of Web Services In: Proceedings of the Workshop on Software Composition (SC 2005), Scotland, April 2005.

[15] Melchiorre, C., Laclavik, M.; Marin, C. ; Carpenter, M. A community based interoperability utility for SMEs. Published in: eChallenges, 2010, 27-29 Oct. 2010.

[16] Oberle, D., Staab, S., Eberhart, A., Semantic Management of Distributed Web Applications, IEEE Distributed Systems Online, vol. 7, no. 4, 2006.

[17] Prinz, W., Loh, H., Pallot, M., Schaffers, H., et al., ECOSPACE -- Towards an Integrated Collaboration Space for eProfessionals. In Proceedings of Collaborative Computing: Networking, Applications and Worksharing, 2006.

[18] Mentzas G., and Friesen A., Semantic Enterprise Application Integration for Business Processes: Service-Oriented Frameworks. 2010.

[19] Popplewell K., Stijanovic N., Abecker A., Apostolou D., Mentzas G., Harding JA. Supporting Adaptive Enterprise Collaboration through Semantic Knowledge Services, in Enterprise Interoperability III: New Challenges and Industrial Approaches, eds. K. Merti, 2008.

[20] Witczynski, M. & Pawlak, A. Virtual Organizations Enabling Net-based Engineering, in Stanford-Smith et al., 908-915, 2002.

[21] Noran, O. Towards a Meta-Methodology for Collaborative Networked Organisations. In: Virtual Enterprises and Collaborative Networks. (Luis M. Camarinha-Matos (Ed.)). pp. 71-78, Kluwer Academic Publisher. 2004.

[22] GloNet Deliverable D3.3 "D3.3 – GloNet Platform V2", October 2013.

[23] Mezgar, I., and Rauschecker, U., The challenge of networked enterprises for cloud computing interoperability. In: Computers in Industry Online28 February 2014.

[24] Wu, D, Greer, M., Rosen, D., Schaefer, D. Cloud manufacturing: Strategic vision and state-of-the-art. In Journal of Manufacturing Systems, Issue 4, pages 564-579, Oct 2013.

[25] Surajbali B., Bauer, M., Bär, H., Alexakis, S.: A Cloud–Based Approach for Collaborative Networks supporting serviced-enhanced products. In *Proceedings of the 14th IFIP WG 5.5 PRO-VE 2013, Sept 30 – Oct. 2*, 2013.

A Secure Mutual Authentication Protocol for Cloud Computing using Secret Sharing and Steganography

Mrudula Sarvabhatla
M.Tech first year, NBKRIST
Tirupathi-517518, A.P
mrudula.s911@gmail.com

M.Giri
Professor and H.O.D
SITAMS, Chitoor,A.P
prof.m.giri@gmail.com

Chandra Sekhar Vorugunti
Dhirubhai Ambani Institute of ICT
Gandhinagar-Gujarat.
vorugunti_chandra_sekhar@daiict.ac.in

Abstract— **Cloud computing is a collection of virtually massive distributed large scale computers to handle enormous enterprise computing, hardware, storage needs. An exponential advancement of communication and information technologies results in substantial traffic for accessing of cloud resources wired and mobile, through various communication devices like desktop, laptop, tabs, etc. via Internet. Significance of enterprise data and increased access rates from low-resource terminal devices demands for reliable and low cost authentication techniques. Lots of researchers have proposed authentication schemes based on password, biometric, steganography etc. with varied efficiencies. In 2014, Nimmy et al proposed a steganography based mutual authentication protocol for cloud computing and claimed that their scheme resists major cryptographic attacks. Unfortunately, in this paper we will show that Nimmy et al is vulnerable to offline password guessing attack and Denial of Service attack. As a part of our contribution, we propose a low cost steganography based authentication scheme which is strongly secure and best suited for asymmetric cloud computing environment.**

Keywords-*Steganography*, *Mutual Authentication, Cloud security, Mobile cloud*

I. INTRODUCTION

Cloud computing is a distributed computing systems architecture, which provides a elastic, on demand computing capacity and virtually infinite database storage capacity through an insecure communication channel called Internet, irrespective of geographical location of client. Due to advancement of communication technologies, exponential number of users are connecting to the cloud servers through light weight resource constrained devices like mobile phone, tab etc. As discussed in [1], the presence of powerful and resourceful machines as cloud servers and a set of resource constrained devices like mobiles, tab etc as terminal devices makes the cloud computing system as asymmetric computing and demands for light-weight cryptography which provides strong security features and minimized processing needs at the terminal side operations.

Many researchers have analyzed security issues focusing on various areas of cloud security. Subhashini et al [2] analyzed security features in cloud management that has originate from the service delivery model (SaaS) of a cloud computing system. Fernandes et al. [3] performed comprehensive analysis on security vulnerabilities, threats and attacks in cloud environment and proposed a classification among them. Aguiar et al. [4] discussed comprehensively on recently discovered attacks on cloud service providers and their countermeasures. The study also covered availability, privacy and integrity, authentication, virtualization, and accountability of client data. Gu et al [5] analyzed security issues in hybrid cloud.

In 2013, Murakami et al [6] have proposed an improvement of security in cloud systems based on steganography. In 2013, Ramachandran et al [7] discussed on security as a service using data steganography in cloud. In 2014, I.M.Khalil [8] analyzed application of data steganagraphy to cloud envinronment. In 2014, Nimmy et al [9] proposed a novel mutual authentication protocol for cloud computing using secret sharing and steganography. They used new features like Out Of Band (OOB) secret sharing. They embed the data to be transferred into an image using Pixel Value Differencing [12]. At server side the image is divided into two shares using (2,2) secret image sharing scheme based on addition [11]. In this paper we will show that Nimmy et al [9] scheme suffers from online password guessing attack and vulnerable to Denial of Service attack. Many researchers [13-16,19] have analyzed that cryptographic and steganographic operations requires huge computational resources and best suitable for resource full server side. In Nimmy et al [9] scheme the resource constrained client side devices are overloaded with huge cryptographic and steganographic operations which makes the scheme impossible to adopt to cloud scenario [1]. As a part of our contribution, we will propose an improved mutual authentication scheme which is light weight on both client and server sides, and resistant to all major cryptographic attacks.

The rest of the paper is organized as follows. In section II a brief review of Nimmy et al's scheme is given. Section III describes the security weakness of Nimmy et al. scheme. In section IV our improved scheme is proposed and its security analyses are discussed in section V. The cost and security comparison of various similar remote authentication schemes are given in section VI and section VII provides the conclusion of the paper.

II. REVIEW OF NIMMY ET AL SCHEME

In this section, we examine the Nimmy et al. [9] novel mutual authentication protocol for cloud computing using secret sharing and steganography. The scheme is composed of four stages: the registration, login, authentication and password change stage. The notations used in Nimmy et al. scheme are listed below:

ID_i: Identity of U_i

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

PW_i: Password chosen by U_i
A.S: Authentication Server.
C.S : Cloud Server.
T_s: A time stamp at which user has sent the login request.
T_i: A secret one time value sent by A.S to U_i using Out of Band channel to U_i's mobile. (T_i value is known only to user and the A.S)
$Share_{U:}$ A secret share given to user by A.S.
$Share_{CS}$: A secret share given to cloud server by A.S.
R_{as}: random value chosen by authentication server. (R_{as} value is known only to A.S).
KAU: A secret key assigned to user by A.S. (KAU is known only to user and the A.S).
KAC: A secret key assigned to cloud server by A.S. (KAC is known only to cloud server and the A.S).
VP: Valid period of the request.
MAC (Message, Key): Message Authentication Code, which computes the message digest based on the key.
h(.): Cryptographic hash function
||: Concatenation
⊕: Bitwise XOR operation.
S: Stegano operation, which converts a cover image to a stego image.
E: Encryption/Decryption operation.

A. Registration Stage

This phase is invoked when a new remote user registers with the cloud server.

(R1) To register himself, U_i selects the identity ID_i and send it to authentication server (AS). AS checks whether ID_i is new or existing. If ID_i is existing, A.S rejects the request else sends an acceptance message to U_i.

(R2) On getting acceptance message, U_i chooses its secret password PW_i and a random number b_i, computes $HU_i = h(ID_i||h(PW_i \oplus b_i))$ and embeds the values into a cover image i.e Stegano{ ID_i, HU_i, T_s} and send it to AS.

(R3) Upon receiving the registration request from U_i, AS proceeds as follows:

a) Extract the message from the stereo Image Extract(Stegano{ ID_i, HU_i, T_s}) to get ID_i, HU_i, T_s.

b) Assign the validation period V.P to the login request.

c) Compute: Secret $S = HU_i \oplus R_{as,}$ $CHU_i = HU_i$.

d) Embed 'S' into a secret image using Pixel Value Differencing [12] and use the image to produce two shares using (2,2) secret image sharing [11] i.e $Share_U$, $Share_{CS.}$

AS issues a smart card to U_i which contains {$Share_u$, VP, CHU_i, ID_i, HU_i, KAU} over secure communication channel. After receiving, U_i stores b_i into smart card memory, where KAU is a secret key assigned by the authentication server AS to U_i. Total contents of S.C = {$Share_U$, V.P, CHU_i, ID_i, HU_i ,KAU,b_i}.

B. Login Stage

This stage will be executed whenever user U_i intends to access the cloud resources.

(L1) U_i submits the smart card (S.C) into the smart card terminal and key in the ID_i^*, PW_i^*.

(L2) S.C proceeds as follows:

Compute: $HU_i^* = h(ID_i^*||h(PW_i^* \oplus b_i^*))$ and compare the computed HU_i^* with the pwc stored in the S.C. if both are equal, S.C authenticates the U_i and proceeds further, else rejects the request.
(L3) S.C computes E_{KAU} {ID_i, T_s} and embeds the same into an image i.e Stegano(E_{KAU}{ID_i, T_s}) and sends the same to A.S.
(L4)A.S extracts the message and decrypts it to get {ID_i, T_s}. A.S generates one time key E_{KAU}{T_i} and sends the same to U_i using Out of Band channel to U_i's mobile.
(L5) On receiving the one time secret key from A.S, U_i submits the same to S.C. S.C decrypts it to get T_i.
(L6) S.C computes:
{$Share_u$ || E_{KAU}{MAC ($Share_{U,}$ T_i),ID_i,T_s,V.P}} and send to A.S.

C. Mutual Authentication Stage

Upon receiving the login request message from smart card, authentication server AS performs following actions:

M1) Decrypt the message part E_{KAU}{MAC($Share_{U,}$ T_i),ID_i,T_s,V.P} to get MAC($Share_{U,}$ T_i),ID_i,T_s,V.P.

M2) Check the validity period V.P if it is valid proceeds further else rejects the login request.
M3) Compute MAC($Share_U$, T_i) and compare the same with the received MAC($Share_U$, T_i). If both are valid proceed further, else rejects the login request.
M4) Compute E_{KAC}{ID_i, T_s} to the cloud server in which the actual enterprise data is stored.

The cloud server (CS), responses to the message by performing following steps:

M5) CS decrypts the message E_{KAC}{ID_i, T_s} to retrieve ID_i, T_s. based on U_i identity i.e ID_i, CS will retrieve the secret share part '$Share_{CS}$' , computes MAC($Share_{CS}$) and frames the reply message {$Share_{CS,}$||E_{KAC}{MAC ($Share_{CS}$, KAC) }}.

On receiving the reply message from CS, AS proceeds as follows:

M6) Decrypt the login reply message to get {MAC ($Share_{CS}$, KAC) }. CS computes MAC ($Share_{CS}$, KAC) and compares with the received value. If both are equal, AS authenticates the cloud server CS else rejects the reply message.

M7) Authentication Server A.S combines $Share_U$ received from U_i and $Share_{CS}$ from cloud server CS to get the secret image.
M8) AS extracts secret of U_i i.e 'S' from the secret image and computes $S \oplus R_{as} = HU_i$.
M9) AS computes session key $S.K = h(HU_i \oplus T_i)$ and sends the encrypted message $E_{KAC}\{ID_i, T_s, S.K\}$ to cloud server CS.

On receiving the reply message from AS, CS proceeds as follows:

M10) Cloud server decrypts to get ID_i, T_s, S.K, computes $E_{S.K}$ $\{ID_i, T_s\}$ and embeds into an image and sends the message $Stegano(E_{S.K} \{ID_i, T_s\})$ to U_i.

On receiving the reply message from CS, U_i proceeds as follows:

M11) Extract the message from the stereo message i.e $E_{S.K}$ $\{ID_i, T_s\}$. Compute $S.K^* = h(HU_i \oplus T_i)$, decrypts the received message i.e $E_{S.K}$ $\{ID_i, T_s\}$. If it contains ID_i, T_s, U_i confirms that the session key is correct and authenticates the cloud server else rejects the message.

On authentication, all further messages are encrypted with the session key framed between user U_i and cloud server CS.

III. CRYPTANALYSIS OF NIMMY ET AL SCHEME

Nimmy et al. [9] proposed a mutual authentication protocol which suits for cloud computing based on secret sharing and steganography and claimed that their scheme is resistance to all major cryptographic attacks. In this section, we will show that Nimmy et al. scheme cannot prevent any of the attacks they claimed that their scheme will resists and still vulnerable to password guessing attack, Denial of Service attack.

Assumptions

While cryptanalysis of Nimmy et al [9] scheme and security analysis of our improved scheme, we will follow the below mentioned well practiced assumptions in the literature [20,21]

1) The adversary is having access to the values stored in the smart card of a legal user.

2) The adversary is having access to all the messages exchanged between the U_i and the servers i.e. authentication server AS and cloud server CS.

A. Stolen Smart Card Attack

A legal attacker 'E', if gets the smart card of any valid user U_i of the system for a while or stolen the card, 'E' can extract the secret data stored in U_i's smart card by some means [17,18]. 'E' can get $\{Share_U, VP, CHU_i, ID_i, HU_i, KAU, b_i\}$ where $CHU_i = HU_i = \{ID_i||h(PW_i \oplus b_i)\}$, after getting these values stored in the smart card of U_i, the attacker 'E' can perform the following attacks as described below.

B. Offline Password Guessing Attack

As discussed above once the adversary 'E' stolen the smart card, 'E' can get the values stored in the smart card i.e $\{Share_U, VP, CHU_i, ID_i, HU_i, KAU, b_i\}$ where $CHU_i = HU_i = \{ID_i||h(PW_i \oplus b_i)\}$. In HU_i, 'E' knows ID_i, b_i. So 'E' can launch password guessing attack on HU_i as follows.

Step 1) Guess the value of PW_i to be PW_i^* from a uniformly distributed dictionary.
Step 2) Compute $HU_i^* = \{ID_i||h(PW_i^* \oplus b_i)\}$
Step 3) Verify whether HU_i^* equals HU_i stored in the smart card.
Step 4) If yes, the password of U_i is PW_i^*, else 'E' can repeat the steps from 1to 3.
On getting the correct ID_i, PW_i, the adversary 'E' can replicate smart card similar to U_i and can execute password change phase of Nimmy et al. [9] scheme successfully, so that the legal user U_i cannot login.

C. Inefficient Password Change Stage

In Nimmy et al [9] scheme the password change stage is as follows:

Step1) U_i provides his smart card into the terminal and key in his identity and password i.e ID_i and PW_i.

Step 2) U_i opts for 'Change Password' option.
Step 3) U_i selects a new password PW_i^{new} , generates a new random number b_i^{new}, and computes $HU_i^{new} = h(ID_i||h(PW_i^{new} \oplus b_i^{new}))$.
Step 4) U_i assigns $pwc_{new} = HU_i^{new}$ and deletes the old values and replace with the new values.

As discussed in (A), the attacker 'E', if gets the smart card of a legal user U_i, 'E' can get the ID_i of a legal user U_i as it is stored in a plain format. Now 'E' can perform the following steps:

Step1) 'E' provides 'U_i' smart card into the terminal and key in U_i identity and a random value as password i.e ID_i and PW_r.
Step 2) 'E' opts for 'Change Password' option.
Step 3) 'E' enters a random value as password PW_r and generates a new random number b_r.
Step 4) S.C computes $h(PW_r \oplus b_r)$.
Step 5) S.C computes $pwc_{new} = h(ID_i||h(PW_r \oplus b_r))$ and deletes the old values and replace with the new values.

Next time when U_i logs into the system, and enters the correct password PW_i, the smart card S.C proceeds as follows:
Compute: $HU_i^{new} = \{ID_i||h(PW_i \oplus b_i)\}$, and compares the computed value HU_i^{new} with the stored value pwc. But the current value of $pwc = h(ID_i||h(PW_r \oplus b_r))$ and definitely not

equal to HU_i^{new} . Hence, the legal U_i never logs into the system again, which leads to Denial of Service (DoS) attack.

Reason for DoS attack: In Nimmy et al scheme the smart card doesn't compute the HU_i with the original ID_i and PW_i values entered by the user and compare HU_i with the pwc value stored in the smart card. Without any cross checking and validating the ID_i and PW_i, S.C reads the new values of PW_i and b_i i.e PW_i^{new}, b_i^{new} values. It directly computes the value pwc_{new} and replaces the old values.

IV. Our Improved Scheme

In this section we describe our improved mutual authentication protocol for cloud computing using secret sharing and steganography over Nimmy et al [9] scheme, which is divided into four stages: Registration stage, Login stage, Mutual authentication stage and Password change stage. Similar to Nimmy et al, we are also adopting the following assumptions:

1) Cloud server, authentication server, users are honest during the registration phase.

2) The registration request is transmitted to the authentication server through a secure channel.

3) Communication channel between the CS and AS is a secure and trustworthy channel.

A. Registration Stage

This stage is invoked when a new remote user registers with the cloud server.

(R1) To register himself, U_i selects his identity ID_i and send it to authentication server (AS). AS checks whether ID_i is new or existing. If ID_i is existing, A.S rejects the request else sends an acceptance message to U_i.

(R2) On getting acceptance message, U_i chooses its secret password PW_i and a random number b_i, computes $HU_i = h(ID_i||h(PW_i \oplus b_i))$ and sends the message $\{ID_i, HU_i, T_s\}$ to A.S through a secure communication channel.

(R3) Upon receiving the registration request from U_i, AS proceeds as follows:

a) Assign the validation period V.P to the login request.

b) Compute: Secret $S = HU_i \oplus R_{as}$, $CHU_i = HU_i$.
Embed 'S' into a secret image and use the image to produce two shares using secret sharing i.e $Share_U$, $Share_{CS}$.

c) Computes: $S_U = Share_U \oplus h(ID_i)$, $HKAU = KAU \oplus h(ID_i)$,

d) AS issues a smart card to U_i which contains $\{S_U, VP, CHU_i, HKAU\}$ over secure communication channel. After receiving, U_i computes $B_i = b_i \oplus h(ID_i||PW_i)$ and stores B_i into smart card memory, total contents of S.C = $\{S_U, VP, CHU_i, HKAU, B_i\}$.

e) A.S sends $\{Share_{CS}, E_{KCS}\{ID_i\}\}$ to the cloud server CS.

B. Login Stage

This stage will be executed whenever user U_i intends to access the cloud resources.

(L1) U_i submits the smart card (S.C) into the smart card terminal and key in the ID_i^*, PW_i^*.

(L2) S.C proceeds as follows:

Compute: $HU_i^* = h(ID_i^*||h(PW_i^* \oplus b_i^*))$ and compare the computed HU_i^* with the CHU_i stored in the S.C. if both are equal, S.C authenticates the U_i and proceeds further else rejects the request.
(L3) S.C retrieves $KAU = HKAU \oplus h(ID_i)$.
(L3) S.C computes $\{ID_i \oplus h(KAU), T_s \oplus h(KAU)\}$, and sends the same to A.S.
(L4) The A.S knows KAU, therefore it retrieves $\{ID_i, T_s\}$. A.S generates one time secret key $\{T_{sec} \oplus h(KAU)\}$ and sends the same to U_i's mobile, using Out of Band channel.
(L5) On receiving the one time secret key from A.S, U_i submits the same to S.C. S.C retrieves T_{sec} from $T_{sec} \oplus h(KAU)$ as it knows KAU.
(L6) S.C computes:

$NID_i = ID_i \oplus h(T_{sec})$.
$M1 = Share_U \oplus h(T_{sec}||KAU)$.
$M2 = MAC(Share_U, T_{sec})$
$T = T_s \oplus h(KAU)$

and sends the login message $\{NID_i, M1, M2, T, V.P\}$ to A.S.

C. Mutual Authentication Stage

Upon receiving the login request message from smart card, authentication server AS performs following actions:

M1) Check the validity period V.P, if it is valid proceeds further else rejects the login request.

M2) Retrieve the secret key sent to U_i mobile through secure Out of Bond channel, i.e T_{sec} and retrieves the following values:

$ID_i = NID_i \oplus h(T_{sec})$.
$Share_U = M1 \oplus h(T_{sec}||KAU)$.

(As authentication server knows both T_{sec} and KAU)

$T_s = T \oplus h(KAU)$.

M3) Check the validity of T_s. If validity fails reject the login request else proceed further.
M4) Compute: $M2^* = MAC(Share_U, T_{sec})$ from the received values and compare $M2^*$ with the received M2. If both are identical then A.S authenticates the U_i else rejects the login request.

On validating the user U_i, the authenticating server gets the second part of secret sharing i.e. $Share_{CS}$ from cloud server.

M5) Compute $E_{KAC}\{ID_i, T_s, T_{sec}\}$ to the cloud server in which the actual enterprise data is stored.

The cloud server (CS), responses to the message by performing following steps:

M6) CS decrypts the message $E_{KAC}\{ID_i, T_s, T_{Sec}\}$ to retrieve ID_i, T_s, T_{sec}. Based on U_i identity i.e ID_i, CS will retrieve the secret share part '$Share_{CS}$' and computes MAC($Share_{CS}$, KAC) and frames the reply message $\{Share_{CS} \oplus h(T_S \| KAC) \| E_{KAC}\{MAC (Share_{CS}, KAC), T_s\}\}$.

On receiving the reply message from CS, AS proceeds as follows:

M7) Decrypt the login reply message to get {(MAC ($Share_{CS}$, KAC) }. AS computes (MAC ($Share_{CS}$, KAC) and compares with the received value. If both are equal, AS authenticates the cloud server CS else rejects the reply message.
M8) Authentication Server A.S combines $Share_U$ received from U_i and $Share_{CS}$ from cloud server CS to get the secret image S.
M9) AS extracts secret of U_i i.e 'S' from the secret image and computes $S \oplus R_{as} = HU_i$.
M10) AS computes session key $S.K = h(HU_i \oplus T_i \oplus T_{Sec})$ and sends the encrypted message $E_{KAC}\{ID_i, T_s, S.K\}$ to cloud server CS.

On receiving the reply message from AS, CS proceeds as follows:

M11) Cloud server decrypts to get ID_i, T_s, S.K and computes $R1=ID_i \oplus h(S.K)$, $R2 = T_s \oplus h(S.K)$ and sends the login reply message {R1, R2} to U_i.

On receiving the reply message from CS, U_i proceeds as follows:

M12) Compute: $S.K^* = h(HU_i \oplus T_i \oplus T_{Sec})$,
Retrieve: $ID_i^* = R1 \oplus h(S.K^*)$
$T_s^* = R2 \oplus h(S.K^*)$
If ID_i^* equals ID_i and T_s^* equals T_s, U_i confirms that the session key is correct and authenticates the cloud server else rejects the message.

On authentication, all further messages are encrypted with the session key framed between user U_i and cloud server CS.

D. Password Change Stage

When a legal user intends to change his password PW_i, U_i provides his smart card to the card reader and the card reader performs the following steps to update the user password:

Step1) U_i provides his smart card into the terminal and key in his identity and password i.e ID_i^* and PW_i^*.

Step 2) U_i opts for 'Change Password' option.
Step 3) Smart card computes $CHU_i^* = h(ID_i^* \| h(PW_i^* \oplus b_i))$. Smart card checks if CHU_i^* computed equal to CHU_i stored in the U_i smart card. If yes, S.C proceeds further, else aborts the process.
Step 3) U_i selects a new password PW_i^{new} and generates a new random number b_i^{new}, and computes $CHU_i^{new} = h(ID_i^* \| h(PW_i^{new} \oplus b_i^{new}))$.
Step 4) U_i deletes the old values and replace with the new values.

V. SECURITY ANALYSIS OF IMPROVED SCHEME

In this section we will discuss and demonstrate how our proposed scheme fixes the vulnerabilities found in Nimmy et al. [9] scheme while preserving the merits of their scheme.

A. Resistance to Stolen Smart Card Attack

A legal attacker 'E', if gets the smart card of any valid user U_i of the system for a while or stolen the card, 'E' can extract the secret data stored in U_i's smart card by some means [3,4]. 'E' can get $\{S_U, VP, CHU_i, HKAU, B_i\}$ where $S_U = Share_U \oplus h(ID_i)$, $HKAU = KAU \oplus h(ID_i)$, $CHU_i = h(ID_i \| h(PW_i \oplus b_i))$, $B_i = b_i \oplus h(ID_i \| PW_i)$. To get $Share_U$ from S_U and KAU from HKAU, the identity ID_i of the user is required. One way of getting ID_i is from NID_i i.e $NID_i = ID_i \oplus h(T_{sec})$ which is a part of login request message, but to get ID_i from NID_i the attacker needs T_{sec}. T_{sec} is known only to the user because it is sent to U_i using secure OOB (Out of Band) channel. Hence T_{sec} is known only to the user U_i. It is clear that the attacker has no way to intercept the ID_i, $Share_U$, KAU,T_s. Hence, in our scheme it's not possible for an attacker to achieve any unknown value from the stolen smart card and the login request message.

B. Resistance to User Impersonation Attack

In order to impersonate a legal user U_i, a legal adversary 'E' must fake a login message {NID_i,M1,M2,T,V.P} to the remote authentication server A.S. To frame the login parameters correctly, 'E' must know ID_i, T_{sec}, $Share_U$, KAU,T_s of U_i. As discussed above it is not possible for 'E' to get ID_i, $Share_U$, KAU,T_s of U_i. Hence, it's not possible for 'E' to compute NID_i,M1,M2,T. Therefore, in our scheme it is impossible for any kind of user to create a fake login message and impersonate a legal user U_i.

C. Offline Password Guessing Attack

As discussed above, if an attacker gets the smart card of a legal user U_i, 'E' can retrieve $CHU_i = h(ID_i \| h(PW_i \oplus b_i))$. CHU_i is the only parameter which contains the password of U_i. To guess the password, 'E' must know ID_i, b_i of U_i. As

discussed in A, it's not possible for 'E' to get ID_i of U_i. To achieve b_i from $B_i = b_i \oplus h(ID_i||PW_i)$, 'E' must know ID_i and PW_i. Hence, it is not possible for an attacker 'E' to perform password guessing attack to achieve the password of U_i.

D. Resistance to Server Masquerade Attack

In order to masquerade as remote server S, an attacker 'E' has to send U_i, a forged reply message { R1, R2} where $R1=ID_i \oplus h(S.K)$, $R2 = T_s \oplus h(S.K)$. To frame S.K, 'E' needs HU_i, T_i, T_{Sec}. 'E' can get HU_i which is equal to CHU_i from U_i stolen smart card, but as discussed above it is not possible to get T_i, T_{Sec}. Hence, it is not possible for an attacker to frame and send the message {R1, R2} to U_i. If an attacker 'E' replays the message as R2 contains T_s which informs the U_i that it is a replayed one. Therefore, in our scheme it is impossible for anyone to masquerade as server.

E. Resistance to Insider Attack

The legal user U_i while registering with the authentication server A.S, sends the registration message $\{ID_i, HU_i, T_s\}$, where $HU_i = h(ID_i||h(PW_i \oplus b_i))$. The insider doesn't know b_i, hence, it is not possible for an insider to compute PW_i. Authentication server and cloud server exchanges the messages $E_{KAC}\{ID_i, T_s, T_{sec}\}$, $\{Share_{CS} \oplus h(T_S||KAC)||E_{KAC}\{MAC(Share_{CS}, KAC), T_s\}\}$.The secret keys KAC, KAU are known only to the A.S, U_i and CS. Hence, it is not possible for an insider to achieve secret keys and decrypt the messages.

F. Resistance to Replay attack

Assume that the attacker intercepted the login communication message sent by U_i to A.S i.e $\{NID_i, M1, M2, T, V.P\}$ where $NID_i = ID_i \oplus h(T_{sec})$. $M1 = Share_U \oplus h(T_{sec}||KAU)$. $M2 = MAC(Share_U, T_{sec}), T = T_s \oplus h(KAU)$. On receiving the login request the authentication server makes an entry T_s, T_{sec} against ID_i. If A.S receives the login request message with $T = T_s \oplus h(KAU)$, It checks for an entry in the database for T_s. If it finds the entry in the database it rejects the request. Therefore, it is not possible for an attacker to simply replay the login messages to A.S. Similarly in the messages exchanged between A.S and C.S i.e $E_{KAC}\{ID_i, T_s, T_{sec}\}$, $\{Share_{CS} \oplus h(T_S||KAC), ||E_{KAC}\{MAC(Share_{CS}, KAC), T_S\}\}$. The time T_s is maintained to restrict the replay attacks by the attackers.

G. Efficient Password Change Stage

In our password change phase the S.C first validates the ID_i and PW_i, by computing $CHU_i^* = h(ID_i||h(PW_i \oplus b_i))$ and comparing the computed CHU_i^* with the CHU_i value stored in the S.C. If both are equal, it confirms that the U_i entered the correct identity and password. Hence, in our scheme only the valid user can update the password. Based on the same discussion we can conclude that this leads to resistance of Denial of Service attack in our scheme.

H. Achieves Strong Mutual Authentication

In our scheme as discussed above its not possible for an attacker to perform user impersonation attack, server masquerade attack and replay the login messages. Hence, we can confirm that our scheme provides strong mutual authentication among the communicating parties.

I. Resists Man in the Middle attack

In our scheme , as discussed in M8, M9 of mutual authentication phase, authentication server A.S, combines the secret share '$Share_U$' received from U_i and secret share '$Share_{CS}$' from the cloud server CS to get the secret image 'S' i.e $S = Share_U + Share_{CS}$. On computing the secret, A.S retrieves HU_i to validate the user U_i. To perform MiM attack, the attacker 'E' has to send both the secret shares of the image 'S'. As discussed above it is not possible for any kind of attacker to get secret share of U_i i.e $Share_U$. Also to get secret share of the cloud server i.e '$Share_{CS}$', 'E' must know Ts, KAC, to get $Share_{CS}$ from $Share_{CS} \oplus h(T_S||KAC)$. As discussed KAC is known only to authentication server A.S and Cloud server C.S. Hence, it is not possble for an attacker to get '$Share_{CS}$'. Without '$Share_{CS}$', 'E' cannot pass through the authentication steps of A.S. Therefore, we can conclude that our scheme resists Man in the Middle attack.

VI. Cost and Security Analysis

Table II, shows the total number of cryptographic operations needs to be performed by both smart card or user side and authentication, cloud server. In our scheme the resource constrained smart card reader need to perform only 20 hash operations and 2 MAC operations, where as in Nimmy et al [9] scheme the smart card need to perform 5h, 3E, 3S, 1M where h: Hash function, E: Encryption or Decryption, S: Stegano operation, M: Keyed hash operations. Our scheme also performs reduced operations even on resourceful server side which results in faster response to the users.

To perform a cryptographic operation like hash, encryption, decryption, stegano, the smart card or server needs to perform certain number of instructions, the instructions associated to a cryptographic operation is discussed clearly in [10]. The summary of discussions from [10] is represented in Table III. Table III represents the number of instructions needs to execute to perform pure steganography, RSA Encryption followed by applying Stegano function, Deffie-Hellman Encryption followed by applying Stegano function. The columns denote how the steganography is performed by replacing 1Least Significant Bit or 2LSB or 3LSB or 4LSB of image with data to be sent.

Table IV denotes the messages which are sent or received by the smart card and number of instructions to be executed by the smart card to perform those operations. It is clear that our proposed scheme is not using any of heavy weight cryptographic operation like Stegano, Encryption and Decryption. Our proposed scheme reducing a total of

63000 instructions to be executed by a smart card when an image Size (64*64) pixels is considered.

In Table I, we have compared the security strength of Nimmy et al [9] and our proposed scheme. Our scheme resists all major cryptographic attacks on which Nimmy et al fails to withstand.

TABLE I. COMPARISON OF SECURITY FEATURES

SECURITY PROPERTIES	OUR's	NIMMY ET AL
Resists stolen smart card attack	Yes	No
Resists password guessing attack	Yes	No
Resists Denial-of-Service attack	Yes	No
Provides early wrong password detection.	Yes	No
Provides early wrong identifier detection	Yes	No

TABLE II. EFFICIENCY COMPARISON AMONG NIMMY ET AL AND PROPOSED SCHEME

PHASE	NIMMY ET AL		PROPOSED	
	Smart Card/User	Server	Smart Card/User	Server
Registration	2h+1S	1S	3h	1h
Login	2h+2E+1S+1M	2E+1S	6h+1M	1h
Authentication	1h+1E+1S	1h+8E+1S+3M	11h+1M	2h+6E+3M
Total	**5h+3E+3S+1M**	**1h+10E+3S+3M**	**20h+2M**	**4h+6E+3M**

h:hash operation, S: Stegano operation, E: encryption and decryption operation, M: Keyed hash i.e MAC.

TABLE III. THE COMPUTATION REQUIREMENT IN TERMS OF INSTRUCTIONS TO EXECUTE EACH OPERATION

Operation	One bit steganography (1LSB of the image is replaced with data)	Two bit steganography (2 LSB of the image is replaced with data)	Three bit steganography (3 LSB of the image is replaced with data)	Four bit steganography (4 LSB of the image is replaced with data)
Pure Steganography	20000/53000 Instructions	35000/55000 Instructions	48000/98000 Instructions	48000 / 98000 Instructions
RSA Encryption + Steganography	52000 /120000 Instructions	100000/150000 Instructions	140000/198000 Instructions	170000 /222000 Instructions
Deffie-Hellman Encryption+ Steganography	21000 /53000 Instructions	40000/56500 Instructions	48000/99000 Instructions	50000/99000 Instructions
	a/b means : 'a' number of instructions need to execute in case the image size is (64*64) pixels / 'b' number of instructions need to execute in case the image size is(128*128) pixels. Image Size (64*64) pixels/ Image Size (128*128) pixels			

TABLE IV. THE NUMBER OF INSTRUCTIONS REDUCED FOR A SMART CARD IN PROPOSED SCHEME COMPARED TO NIMMY ET AL.

Message Sent in Nimmy et al scheme	Number of Instructions to be executed by S.C for {S, E, M}	Message Sent in our scheme	Number of Instructions to be executed by S.C for {S, E, M}
Stegano$\{ID_i, HU_i, T_s\}$ in (R2)	20000	$\{ID_i, HU_i, T_s\}$ in (R2)	0
Stegano$(E_{KAU}\ \{ID_i, T_s\})$ in (L4)	21000	$\{ID_i \oplus h(KAU), T_s \oplus h(KAU)\}$ in (L3)	0
$E_{KAU}\{MAC\ (Share_u, T_i), ID_i, T_s, V.P\}\}$ in (L6)	1000	$\{NID_i, M1, M2, T, V.P\}$ in (L6)	0

Extract and Decrypt of : Stegano ($E_{S.K}$ {ID_i, T_s}) in (M11)	21000	{R1, R2} in (M11)	0
Total	63000	Total	0

VII. CONCLUSION

In this manuscript, we have cryptanalysed the mutual authentication protocol for cloud computing using secret sharing and steganography proposed by Nimmy et al [9]. We have shown that Nimmy et al scheme is vulnerable to various attacks and there is a scope for reducing the heavy weight cryptographic operations on resource drained client side. We have proposed our improved scheme in which, we taken off the resource consuming stegano, encryption and decryption operations from client side. In our scheme the load on the servers are reduced drastically which results in faster response to users from server side. We analyzed security strength of our scheme and shown that it is resistant to all major cryptographic attacks. Less computation requirement from client side and resistant to all major cryptographic attacks makes our scheme more practical and adoptable to use even with resource constrained devices like mobile, tabs etc.

REFERENCES

[1] V. K. Zadiraka and A. M. Kudin, "Cloud computing in cryptography and steganography", springer journal of Cybernetics and Systems Analysis, Volume 49, Issue 4, pp 584-588,July 2013.

[2] S. Subashini, and V.Kavitha, "A survey on security issues in service delivery models of cloud computing", Elsevier Journal ofNetworkandComputerApplications, vol 34, pp: 1–11, 2011.

[3] D.A.B. Fernandes,L.F.B. Soares,J.V. Gomes, M.M.Freire, and P. R. M. Inácio , "Security issues in cloud environments: a survey", springer international Journal of Information Security,Volume 13, Issue 2, pp 113-170, April 2014.

[4] E. Aguiar,Y.Zhang and M.Blanton, "An Overview of Issues and Recent Developments in Cloud Computing and Storage Security", springer journal of High Performance Cloud Auditing and Applications pp 3-33, 2014.

[5] Q. Gu, and M. Guirguis. "Secure Mobile Cloud Computing and Security Issues", Springer: High Performance Cloud Auditing and Applications, pp 65-90, 2014.

[6] K.Murakami,K. Hanyu, R. Q. Zhao, and Y.Kaneda,"Improvement of security in cloud systems based on steganography", International Joint Conference on Awareness Science and Technology and Ubi-Media Computing, pp:503 - 508, 2-4 Nov. 2013.

[7] A.B. Ramachandran, P.Pradeepan, and M.Saswati," Security as a Service using Data Steganography in Cloud", The International Conference on Cloud Security and Management , Washington University, Seattle, USA,Oct 17-18,October 2013.

[8] I.M.Khalil, A.Khreishah, M.Azeem, "Cloud Computing Security: A Survey", MDPI: Computers, vol 3, pp:1-35, 2014.

[9] K.Nimmy, M.Sethumadhavan, "Novel mutual authentication protocol for cloud computing using secret sharing and steganography", Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT), feb, 2014, India.

[10] G.Shailender, G.Ankur, B.Bharat, "Information Hiding Using Least Significant Bit Steganography and Cryptography",International Journal of Modern Education and Computer Science,Vol6,27-34, 2012.

[11] N. Chen and R. Jiang, "Security Analysis and Improvement of User Authentication Framework for Cloud Computing", Journal of Networks, pp.198-203, 2014.

[12] L. Dong and M. Ku, "Novel (n, n) secret image sharing scheme based on addition," in Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), Sixth International Conference on, pp. 583–586. 2010.

[13] S. Kondo and Q. F. Zhao, "A novel steganographic technique based on imagemorphing," Proc. International Conference on Ubiquitous Intelligenceand Computing, Lecture Notes in Computer Science 4159, Springer, UIC06, pp. 806-815,Sept. 2006,China.

[14] S. Katzenbeisser and F. A. P. Peticolas, Information hiding: techniques for steganography and digital watermarking, Artech House, 2000.

[15] N.Provos, and P.Honeyman, "Detecting steganographic content on the Internet",Proceedings of Network and Distributed System Security Symposium. Internet Society, Reston, VA, (San Diego, Feb. 6–8), 2002.

[16] M.Bevilacqua, A.Roumy, C.Guillemot, and M.L.Alberi More, "Low-Complexity Single-Image Super-Resolution based on Nonnegative Neighbor Embedding," in BMVC (British Machine Vision Conference), 2012.

[17] T.S.Messerges, E.A.Dabbish and R.H.Sloan, "Examing smartcard security under the threat of power analysis attacks," IEEE Transactions on Computers, vol. 5, no. 3, pp. 514-522, 2002.

[18] E. Brier, C. Clavier, and F. Oliver, "Correlation power analysis with a leakage model ", Lecture Notes in Computer Science, Vol. 3156, 2004, pp. 135-152.

[19] V. K. Zadiraka and A. M. Kudin, "Features of the implementation of cryptographic and steganographic systems based on the cloud computing principle," Iskusstv. Intellekt, No. 3, pp. 438–444, 2012.

[20] J.M. Kim, and J.K. Moon, "Approach of Secure Authentication System for Hybrid Cloud Service", Advanced in Computer Science and its Applications, LNEE Vol 279, pp 543-550, 2014.

[21] B.K. Chaurasia, A. S. hahi, and S. Verma, "Authentication in Cloud Computing Environment Using Two Factor Authentication", Proceedings of the Third ICSCPSAISC Vol 259, pp 779-785,2014.

Analyzing User Behavior Using KeyStroke Dynamics to Protect Cloud from Malicious Insiders

Mahesh Babu B.
Department of Computer Science and Engineering
National Institute of Technology, Tiruchirappalli
Tiruchirappalli 620015, Tamil Nadu, India
Email: 306112003@nitt.edu

S. Mary Saira Bhanu
Department of Computer Science and Engineering
National Institute of Technology, Tiruchirappalli
Tiruchirappalli 620015, Tamil Nadu, India
Email: msb@nitt.edu

***Abstract*—Nowadays cloud computing is growing vastly as the number of companies are depending mostly on this technology due to its advantages. All the data are stored across the globe and maintained by the cloud service providers i.e., the responsibility of the data is in the hands of cloud providers. The data breach is a biggest problem in cloud as the data are shared across the globe and sensitive information of the customer is stored at some third party storage site. Security of the data is the major concern in cloud from outsiders as well as insiders. Insider attack is the most devastating threat due to the familiarity of the underlying system to the insiders. The proposed approach mitigates this threat by a host based user profiling technique where a key stroke dynamics is used for analyzing the user behavior and a retraining approach is also proposed as the imposter patterns are absent at the time of registration. Retraining boosts the overall system performance in mitigating the insider threat.**

Keywords—Cloud, Keystroke dynamics, Security, Insider threat.

I. INTRODUCTION

Cloud computing is a general term for anything that involves delivering hosted services over the Internet and managed by the cloud service provider. The increased use of cloud raises the privacy concerns. Security of the data became the major issue as the data resides outside the user premises and managed by a third party [1]. The insider attack is most devastating compared to an outsider and difficult to find as the complete knowledge of the underlying system is well known to the attacker. An Insider is the one who is having the authorized privileges within the organization. A malicious insider can create more damage to the cloud provider as well as to the users by stealing the sensitive information or by a data breach both to the name and fame [2]. For example, a cloud administrator can access all the virtual machines of the users and can steal the sensitive information of the users without their intervention. A variety of risks has been identified in [3, 4]. Identity Management Systems (IDM) and Intrusion Detection System (IDS) are considered as useful tools for taking proactive measures against insider attacks [10]. They can detect abnormal actions like packets containing malicious or unnecessary content and deviations in normal user behavior. IDS are of two types namely Host based IDS and Network based IDS which are used to isolate a host and track them. The tracking will be done either through available signatures or through anomaly using machine learning techniques by tracking the behavior of the attacks. The proposed work follows a host based user profiling technique to trace the user's behavior using a keystroke dynamics.

One observation made in cloud is that most of the administration work involves command line interface rather than graphical user interface. Since the command line interface requires lot of key strokes, the proposed approach is well suitable for this environment. If the abnormality in the user behavior is detected, the system is locked so that a malicious masquerader cannot do any modification in the name of others.

Keystroke dynamics is a science of studying about keystrokes that differentiate each user based on their typing speed, latency between keystrokes, and pressure applied on keys etc. Key stroke dynamics fall under non-static biometrics which will vary with time. Non-static biometrics depends on several environmental, physical and biological factors [5]. In contrast IRIS, finger prints, palm print etc comes under static biometrics which stays constant for longer duration but they require extra hardware to achieve it, which is not possible in a cloud based environment [6]. To deal with the non-static biometric nature, different features are to be evaluated to attain proper results.

The proposed work uses a Support Vector Machine (SVM) which is one of the best known classifications and regression algorithm to date. Support Vectors (SV) that fall under different regions are separated using hyper planes linear as well as non-linear, and are achieved by adjusting different kernel functions. Researchers have proved that SVM will converge to the best possible solution in very less time. In this work a variant of the SVM called online SVM is used where the results are processed on the fly [31].

There are two important factors that need to be considered in any biometric system. They are False Acceptance Ratio (FAR) and False Rejection Ratio (FRR) that determine the effectiveness of the biometric systems.

FAR is defined as the ratio of the number of times imposter gets accepted to the total number of attempts.

FRR is defined as the ratio of the number of times true user gets rejected to the total number of attempts.

The rest of the paper is organized as follows: section II describes the related work done, section III deals with the

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

proposed keystroke analyzer and its explanation, section IV illustrates the experimental results and setup and finally section V concludes the work.

II. Related work

Mitigating an insider threat is an important research area where many aspects of security such as authentication, authorization, digital forensics, secure data storage as well as legal challenges have been focused. Insider threat exists traditionally for many years and the factors that are considered into account are the psychological and behavioral conditions of the employees.

Salvatore et. al.,[7] proposed an approach to mitigate the insider threat by generating the fog or decoy technology when unauthorized access is suspected then verified using a challenge question. Rocha and Correia [8] describe how easy for the malicious insider to steal the sensitive information. Claycomb, William R., and Alex Nicoll [9] discuss the different levels of administrators and the hierarchy of threat assigned with each of them. Different kind of attackers and attack surfaces used for insider threat are discussed in [10].

Psychology and Sociology are useful tools in the battle against the insider threat. They offer precious information about the motives and the process of a potential insider attack [11]. The CMO model proposed in [12] describes that to claim a person as attacker the necessary conditions are he/she must possess the capacity to do the attack, a motivation and an opportunity to deploy the attack. Some techniques like Proactive analysis [13], graph-based analysis [14], and honey pots [15] had been proposed to detect the insider threat mostly by analyzing user behavior. IDS can be considered as the useful tool in the process of insider detection [16] as they can detect abnormal actions, packets with illegal content and deviations from normal user behavior. Some of the useful approaches to mitigate the insider threat are system call analysis [17] command sequence and usage events tracking [18] and the methods based on usage habits of the users which are called user profiling techniques. A procedure to monitor file system, memory, I/O and hardware had been proposed, to detect differentiation from the normal usage norms that can be used to detect usage anomalies or even confirm the stated level of knowledge of the user along with similar metrics and sophistication [19, 20, 21]. Analytical description of the actions of a malevolent insider has been presented in [22] such that the actions are detected and tie them to the suspicious insider. The proposed work will fall under the same category of behavior analysis that uses an approach called keystroke dynamics.

Several approaches are proposed in keystroke analysis which is in the field of advanced authentication. The prime benefits of keystroke dynamics is low dependency on the dedicated hardware infrastructure [23] and the work mainly focuses on the widely used QWERTY key board or built in laptop keyboard. Wang, Xuan, Fangxia Guo, and Jian-feng Ma [24] uses a set of orthogonal bases and a common feature vectors that are periodically generated from keystroke features of a legitimate user's several recent successful authentications. The current keystroke vectors are then projected onto the set of orthogonal bases to obtain the distribution of the feature vector between its projections. Many of the keystroke dynamic procedures limited themselves to specific part of the keyboard or particular phrases [25]. The imposter key stroke patterns are available only after certain failed login attempts. A retrain frame work is proposed by Lee, Hyoung-joo, and Sungzoon Cho [26] in which linear vector quantization technique is used to retrain the Support Vector Data Description (SVDD) with imposter patterns.

All of the previous work limit to single sequence (eg. Password) which is referred as password hardening. But in a real time scenario password hardening alone is insufficient to identify insiders. So a continuous keystroke approach is needed which will monitor the system continuously rather than only at the time of authentication. The continuous key stroke approach requires considering multiple sequences such as most frequently used English words or most frequently used commands. Frequency of a sequence is measured by a factor called Universality (U) which is defined as the ratio of sequence used by number of users to the total number of users [28]. Bours and pattrick [27] proposed a Continuous Keystroke Dynamics (CKD) system which will continuously monitor the typing behavior of a user. The CKD system will determine if the current user is still a genuine one or not and suitably lock according to the trust level. The proposed approach uses a retraining method along with SVM to overcome the above said limitations and to improve the overall system performance.

III. Proposed approach

The proposed Key Stroke analyzer uses a host based user profiling technique where the monitoring subsystem is a key logger installed in each and every virtual machines and the information is gathered from both the hosts and virtual machines at regular intervals. It resides above the core layer and monitors all the users' interaction with the core and allows only the authorized user (Fig. 1). The abnormality in the behavior will raise alarm and locks the current session.

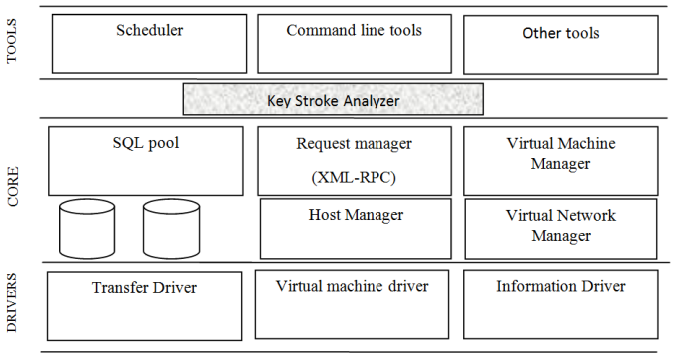


Fig. 1. Key Stroke Analyzer and open nebula architecture

The proposed approach contains three phases: 1) Registration phase, 2) Validation phase and 3) Retraining phase.

A. Registration phase

In the registration phase, user's behavior is monitored for 3 or 4 days and data are collected in the form of key strokes along with timestamps using the previously installed key logger, assuming that in these 3 or 4 days no one else used his/her system and all the keystrokes are user specific. Data

collected from the user are stored in the data store of the cloud in a timely basis. The raw data that are collected are to be processed and are arranged as a set of features before providing them as SVs to the SVM as described in processing raw data. SVM will generate a model file which will act as a user template to validate the user as shown in Fig. 2.

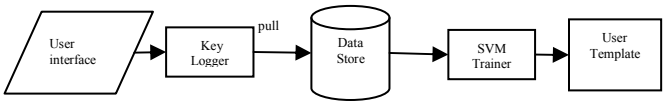
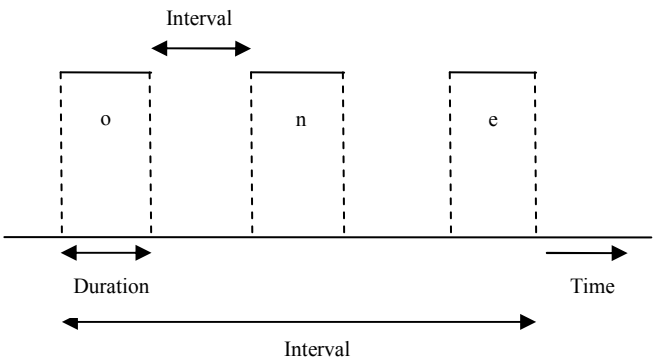


Fig.2. Registration Phase

1) Processing raw data

Raw data collected from the user contain keys and their time stamps only. This data need to be processed before submitting it to the training, verification or retraining phases, and are to be organized into different features.



Number of timing vectors formed:

Key	Press	Release
O	6822999	6974999
N	7055000	7198999
E	7270999	7423000

Duration $= Release_i - Press_i$

Interval $= Press_j - Release_i$
$= Press_j - Press_i$
$= Release_j - Release_i$
$= Release_j - Press_i$

where, i,j represents first and last letters.

$O_{duration}$ = 152000 (Release (O)→Press(O))

$ON_{interval}$ = 80001 Press (N)→Release (O))
= 232001 (Press (N)→Press (O))
= 224000 (Release (N)→Release (O))
= 376000 (Release (N)→Press (O))

$ONE_{Interval}$ = 296000 (Press (E)→Release (O))
= 448000 (Press(E)→Press(T))
= 448001 (Release(E)→Release(T))
= 600001 (Release(E)→Press(T))

Fig. 3. Features evaluation of word 'ONE'

The data containing extreme time stamp values are removed as they arise when switching the application or doing some other work or else due to false presses of the keys.The extreme values are the values that are either too high or too low. The classification is done using a TRIE data structure to separate the data into specified group of words. The words are defined well before and contains frequent letter combinations or sequences in English (e.g. THE, th, ly etc.) a several trigraphs and digraphs, quad-graphs as well as the most frequently used commands (e.g. cmd, one vm, one host etc,)and some of the search terms.

Keystroke activity generates hardware interrupts that can be time stamped and measured up to microseconds (µs) precision and even less. A keystroke activity usually consists of two actions - key press and key release which can be collected along with their time stamps. The following features can be extracted using the key press and key release values.

Dwell Time (DT): Dwell time refers to the amount of time between press and release of a single key, also called duration time and hold time and can be calculated as

$DT_n=R_n-P_n$, where R is the time stamp of key release and P is the time stamp of key press

Flight Time (FT) or Interval: Flight time refers to the amount of time elapsed between pressing and releasing of two successive keys, also called latency time or inters key time or interval time. It involves key events from both the keys. FT can be calculated in four different forms.

$FT_{nPP}= P_{n+1} - P_n$, (Press → Press)

$FT_{nPR}= P_{n+1} - R_n$, (Press → Release)

$FT_{nRP}= R_{n+1} - P_n$, (Release → Press)

$FT_{nRR}= R_{n+1} - R_n$, (Release → Release)

In Di-graph the timing information of two consecutive key-strokes calculated as DT and FT (4 types).

In N-graph the timing information of the first and last key of the sequence, calculated as DT and FT (4 types)

Fig.3. shows a feature generation of a keyword 'ONE', the three level of feature generation has been showed here where the duration refers to single key interval which is the timing difference between same key press and release as shown in Fig.3, the second level of features shown refers to $ON_{interval}$ is the difference between the N and O, there four features generated here press-press, press-release, release-press, release-release. One observation here is the press-press and release-release will have less difference, release-press is the highest value and press-release is the least value among four features. The third level of features generated shown refers to $ONE_{Interval}$ is the difference between the E and O, also generates four features as shown in Fig.3. In this paper a total of 100 keywords are used including all alphabets, Backspace, Shift, Return,10 digits, two letter, three letters and four letters, different commands.

B. Validation phase:

In the Validation phase the user is monitored continuously and data collected is pulled to the cloud through SSH, the raw data is processed as described in the processing raw data section and compare with previously generated user template using SVM prediction method. If the current vector matches with the user template then the user is accepted by the system as true user and the data is stored in a separate data store by setting the access field to 1 for retraining as shown in Fig.4. Otherwise the vectors can be either distorted or imposter patterns. Rejecting the user immediately will raise FRR which will degrade the overall system performance. A trust factor is defined to manage FAR and FRR properly through a reward-penalty function.

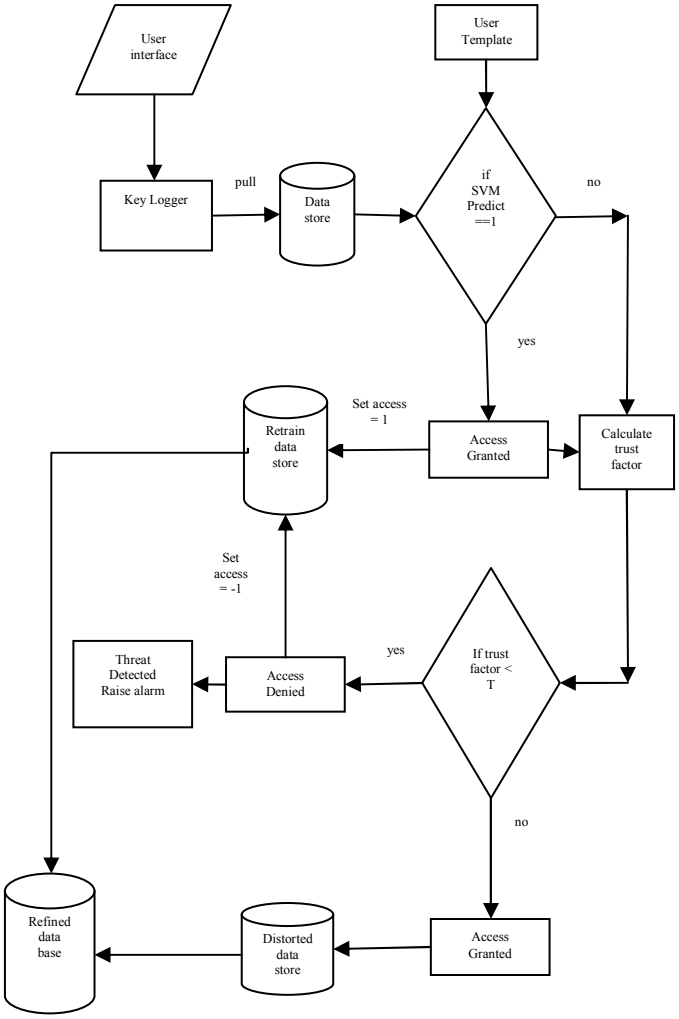


Fig. 4. Validation Phase

1) Trust factor:

Trust Factor (TF) is one of the important factor in the continuous authentication system as the user cannot be rejected on a single failure which will increase the False Rejection Ratio (FRR). User should be validated using some factor so that the true user should not get rejected all the time and the imposter should not take over the system. TF is important because of the distortions in the user behavior in non-static biometric system. The distortion in the user behavior may happen because of several environmental, physical and mental conditions, the type of instrument used or application he is using. TF is a reward-penalty function which will increase if there is match and will decrease if there is a mismatch. The reward-penalty function should be chosen wisely such that FAR and FRR should be minimum. In a continuous key stroke system, it is important to be to cleverly reject the imposter after few keystrokes rather rejecting immediately.

The reward-penalty function is of two types 1) fixed and 2) variable. In fixed reward-penalty the reward and penalty will vary in same ratio and in variable penalty the reward and penalty values will vary by a fraction [29]. The penalty method

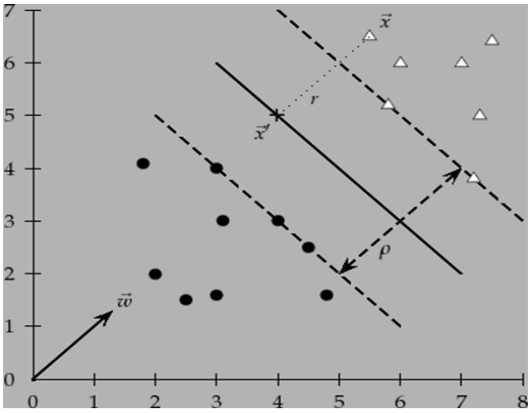


Fig. 5. Hyper planes separating the SVM

followed in keystroke analyzer is a logarithmic barrier function which is represented by

$$g(T, D) = \begin{cases} -\log(T - D) & when\ D < T \\ \infty & Otherwise \end{cases} \quad (1)$$

To minimize a function f(D) using logarithmic barrier function the variable D can be constrained to be strictly lower than some global or local threshold (T). In this work the local thresholds of each individual is adjusted separately by comparing user keystrokes against his/her own template so as to maintain minimum FRR values.

The Distance D is calculated between the SV's and hyper plane using the method defined in [29]. As shown in Fig. 5, D is the shortest distance between a point and a hyper plane which is perpendicular to the plane and hence parallel to a hyper plane ($\vec{W}$). The unit vector in this direction is calculated as $\frac{\vec{w}}{|\vec{w}|}$. The translation of the vector can be obtained by $D(\frac{\vec{W}}{|\vec{W}|})$. Let us label the closest point in the hyper plane to $\vec{X}$ as $\vec{X}'$ then

$$\vec{X}' = \vec{X} - yD\left(\frac{\vec{W}}{|\vec{W}|}\right) \quad (2)$$

$'y'$ changes the sign for two cases of $\vec{X}$ being on either side of the decision surface and should be always need to be +1 as this approach needs to calculate the distance from the positive hyper plane, then from (1)

$$\vec{X}' = \vec{X} - D\left(\frac{\vec{W}}{|\vec{W}|}\right) \quad (3)$$

Since $\vec{X}'$ lies on the decision boundary, it satisfies the equation

$$\vec{W}^T\vec{X}' + b = 0 \quad (4)$$

From (3) and (4)

$$\vec{W}^T\left(\vec{X} - D\left(\frac{\vec{W}}{|\vec{W}|}\right)\right) + b = 0 \quad (5)$$

$$D = \frac{\vec{W}^T\vec{X}+b}{|\vec{W}|} \text{ as } \vec{W}.\vec{W}^T = 1 \quad (6)$$

Now TF can be calculated follows (1) as

$$TF = \begin{cases} TF - \log(T - D) \text{ if match } != \text{true user(penalty)} \\ TF + \log\left(\frac{2}{|w|}\right) \text{ match } == \text{true user(reward)} \end{cases} \quad (7)$$

The reward is defined as $\log\left(\frac{2}{|w|}\right)$ because the two hyper planes can get separated by optimal distance of $\left(\frac{2}{|w|}\right)$. Thus the penalty will always remain greater than or equal to $\log\left(\frac{2}{|w|}\right)$ when there is a mismatch and the reward function will fit well in this range.

TF should always be greater than that of the local threshold otherwise the user will get denied. There are two possibilities when there is a mismatch in the current vector and user template 1) user get accepted based on the TF value and 2) user get permanently denied. In case 1 the user is allowed to access the system but the vectors are stored in distorted data set as there is no match between the user template and the current vector. In case 2 if TF drops below the local threshold then the user will get rejected and denial patterns are stored in a database by setting their access value as -1 which are considered strongly negative for retraining purpose. The collection of imposter patterns and retrain them will improve the overall system performance. The distorted data store contains both weak positive and weak negative patterns. Weak positive patterns are those which are nearer to the positive hyper plane whereas weak negative patterns are away from the positive hyper plane. The distorted data store is considered as strong negative if the user get rejected and unable to provide the adequate credentials to unlock. If the user can prove himself as legitimate then the distorted data store will processed and the SVs nearer to the refined data base hyper plane are added and remaining are discarded. The collected data in retrain data store are added to the refined data store at regular intervals and the refined data store is forward to the retraining phase to get a modified model file which can be used as user template for the subsequent processing of user's behavior.

C. Retraining phase

The retraining phase is same as of validation phase but the efficiency of the retraining lies in its capacity to retrain the new data by modifying the existing model file. An online SVM algorithm [31] can obtain this by altering the direction searches and will converge to the known SVM solution. The advantages by using this method in the keystroke analyzer are

- The training timing required is optimized as Support Vectors (SV) can be divided into sub vectors and trained through break points.
- The algorithm will update the samples only when there is a violation in Karush-Kuhn-Tucker (KKT) conditions [33] until the new data point satisfies the optimality condition.
- Online algorithms require very less memory as the vectors are processed one by one and are discarded after examination.

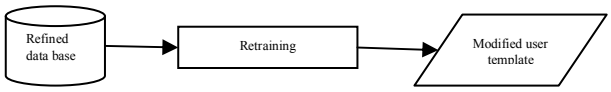


Fig. 6. Retraining Phase

The online SVM algorithm involves two methods 1) process and 2) reprocess which are called as direction searches. The process method checks for potential coefficients which are then become support vectors if exists. In reprocess method the SVs which are making the coefficients to zero are removed from the kernel expansion. By interchanging this direction searches one after the other, this process will converge to the known SVM solution.

The retraining algorithm is defined as the set of successive process and reprocesses methods. The process method resembles Sequential Minimal Optimization method used in LIBSVM [31, 32].

Retraining phase:

1) Retraining:
   a) Repeat a predefined number of times:
      i. Pick an example.
      ii. Run PROCESS.
      iii. Run REPROCESS once.
2) Finishing:
   a) Repeat REPREOCESS until δ<=□ where δ refers to maximal gradient and □ is measure to calculate the KKT conditions.

A pair (i, j) is said to be □-violating pair if and only if it satisfies the following conditions:

- $\alpha_i < B_i$
- $\alpha_j < A_j$
- $g_i - g_j > \square$.

The most important parameters are to be considered in process and reprocess steps are:

S - set of potential vectors indices

α_i - basic coefficient of the current kernel

g_i - partial derivative

The variables α_i and g_i contain meaningful values for all values of i belonging to S. α_i is assumed to be NULL if i does not belong to S and there are some indices belonging to S for which α_i can be zero.

PROCESS:

1) Insert i∉S into S.
2) Search for other index in S to find the □-violating pairs with maximal gradient δ.
3) Go to reprocess if no samples available or else repeat the process of remaining samples.

REPROCESS:

1) Search for □-violating pairs of elements of S with maximal gradient δ.
2) Perform a direction search.
3) Remove non support vectors.
4) Compute bias term ′b′ of the decision function and gradient ′δ′ of the most □-violating pairs in S.

The process of machine learning cannot be applied to raw data which is unstructured. A structural lay out need to be setup where the raw data is arranged into set of features that are predefined.

IV. EXPERIMENTAL SETUP AND RESULTS

Ubuntu 11.04 OS and Open nebula 4.0.1 as the cloud environment were used as the platform having 4GB RAM and 2.1GHz intel core i3 processors for conducting the entire test bench. XINPUT is used as the base for key logger environment and key maps are obtained from XMODMAP for Ubuntu. For windows python packages pyHook and pyWin32 are used. LASVM is used as the classifier and is an online SVM algorithm.

Processing raw data of english alphabets using trie data structure takes O(k*26) where k is the length of keywords. In this work, the order of keywords considered is maximum of 4 and so the complexity becomes O(1). LASVM takes $O(n^3)$ time complexity to classify and predict the user behaviour. Calculation of trust factor also takes O(1) time complexity. So the overall time complexity of the analyzer is $O(n^3)$ where 'n' is the amount of data to be processed. In this paper, for every 1000 keystrokes, user data is processed. The average typing speed on computer is 41 words per minute [34] and the word size is considered or standardized as 5 letters or keys. For 1000 keystrokes, typing takes on an average of 5 minutes duration. For 2 GHz processor it requires 5 seconds to process 1000 keystrokes. So it can process 60 users approximately. If two users having same level of typing proficiency, the analyzer will predict the malicious insider as matching of keywords is considered in addition to typing speed.

The threshold of the individual user is defined based on the user's own data against user template and the imposter data. For the testing purpose an imposter pattern was introduced to the analyzer after valid user has been registered. The imposters are none other than the co-workers of the user as it is for the case of insider attack as shown in Table I.

In keystroke dynamics it is important to detect the imposter as quickly as possible before doing any possible harm rather blocking him immediately, this metric can be measured in number of keys typed before being caught before the TF drops below threshold T. Different combinations are checked against each individual with the different imposters patterns between a set of 4 users as an example.

TABLE I. IMPROVEMENT IN ACCURACY AFTER RETRAINING

User ID	Imposter ID	Number of keys used by imposter before being caught	
		Before retraining	*after retraining*
1	2	123	74
1	3	249	63
1	5	200	85
2	1	97	77
2	3	100	85
2	5	244	124
3	1	271	64
3	2	44	50
3	5	500	63
5	1	120	77
5	2	230	65
5	3	489	100

Table 1 contains 4 columns user id, imposter id, how many keystrokes the imposter get detected before being caught and the number of keystrokes after retraining. The first column defines the true user pattern, second column represent the potential attacker, the third column gives the analyzer capability to detect the imposter and fourth column also gives the analyzer capability to detect the imposter after retraining. Table 1 show that the performance of the system has increased with successive retraining and the mean of used keys before retraining is around 223. After retraining the mean of keys being typed has decrease to 76 which was around 65.91% improvement on an average.

Fig.7 shows that how the keystroke patterns vary for normal, distorted and imposter patterns at different key words for a particular user and for different sequences which will show how the patterns will vary for a particular user at different situations.

The statistical performance of the biometrics is differed for different thresholds or a threshold for each user. The user specific threshold is the threshold minimizing corresponding to its error computed by using own captures for the FRR and imposters captures for the FAR.

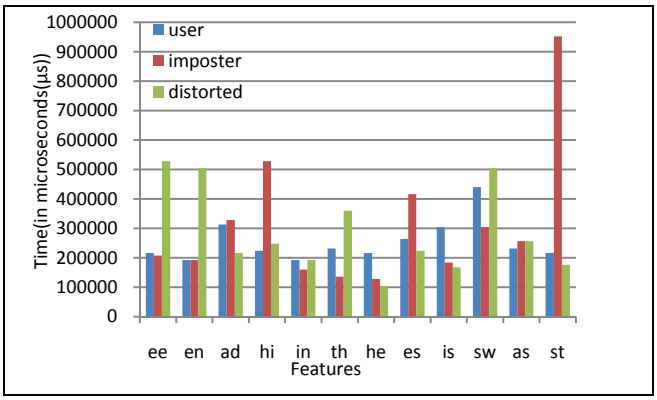


Fig. 7. patterns of a particular user

The individual threshold is measured by comparing the user's vectors against their very owns templates. The threshold which gives suitable FRR will be fixed as local threshold.

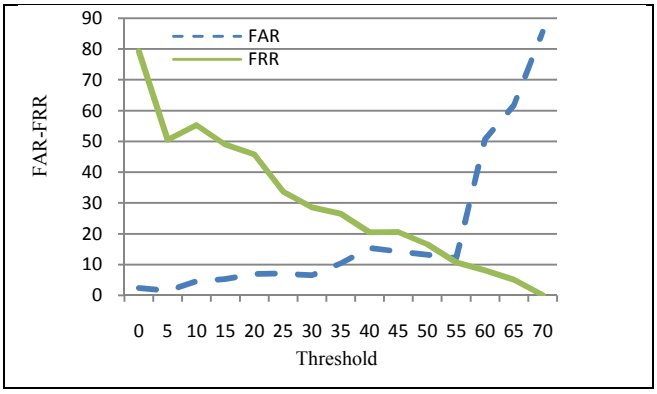


Fig. 8. FAR- FRR values at different Thresholds

A plot shown in Fig.7 was drawn using the average taken from each individual FAR and FRR by adjusting threshold at different levels the FAR-FRR values are at minimum at threshold level of around 55. The proposed approach prevents the masqueraders, as well as it provides authentication.

V. CONCLUSION

In cloud the insider threat has continued to be the biggest problem to date. The proposed work shows better results in mitigating the insider threat in the presence of a masquerader and as well as provides authentication to the user. In addition, the proposed approach does not require any extra hardware and there is no need of any modification in the existing cloud infrastructure for implementation. The retraining results in average increase of 65.91% on the number of keys imposter typed before being caught. In future, the efficiency of the proposed approach will be increased by combining it with the other behavioral techniques like search and command sequence analysis.

REFERENCES

[1] Mell, Peter, and Tim Grance. "Effectively and securely using the cloud computing paradigm." *NIST, Information Technology Lab* (2009).

[2] Rocha, Francisco, and Miguel Correia. "Lucy in the sky without diamonds: Stealing confidential data in the cloud." *Dependable Systems and Networks Workshops (DSN-W), 2011 IEEE/IFIP 41st International Conference on*. IEEE, 2011.

[3] Brunette, Glenn, and Rich Mogull. "Security guidance for critical areas of focus in cloud computing v2. 1." *Cloud Security Alliance* (2009): 1-76.

[4] Heiser, Jay, and Mark Nicolett. "Assessing the security risks of cloud computing." *Gartner Report* (2008).

[5] Teh, Pin Shen, Andrew Beng Jin Teoh, and Shigang Yue. "A survey of keystroke dynamics biometrics." *The Scientific World Journal* 2013 (2013).

[6] Ngugi, Benjamin, Beverly K. Kahn, and Marilyn Tremaine. "Typing biometrics: impact of human learning on performance quality." *Journal of Data and Information Quality (JDIQ)* 2.2 (2011): 11.

[7] Stolfo, Salvatore J., Malek Ben Salem, and Angelos D. Keromytis. "Fog computing: Mitigating insider data theft attacks in the cloud." *Security and Privacy Workshops (SPW), 2012 IEEE Symposium on*. IEEE, 2012.

[8] Rocha, Francisco, and Miguel Correia. "Lucy in the sky without diamonds: Stealing confidential data in the cloud." *Dependable Systems and Networks Workshops (DSN-W), 2011 IEEE/IFIP 41st International Conference on*. IEEE, 2011.

[9] Claycomb, William R., and Alex Nicoll. "Insider threats to cloud computing: Directions for new research challenges." *Computer Software and Applications Conference (COMPSAC), 2012 IEEE 36th Annual*. IEEE, 2012.

[10] Duncan, Adrian J., Sadie Creese, and Michael Goldsmith. "Insider attacks in cloud computing." Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on. IEEE, 2012.

[11] Theoharidou, Marianthi, et al. "The insider threat to information systems and the effectiveness of ISO17799." *Computers & Security* 24.6 (2005): 472-484.

[12] Schultz, E. Eugene. "A framework for understanding and predicting insider attacks." *Computers & Security* 21.6 (2002): 526-531.

[13] Bradford, P., and Ning Hu. "A layered approach to insider threat detection and proactive forensics." *Proceedings of the Twenty-First Annual Computer Security Applications Conference (Technology Blitz)*. 2005.

[14] Eberle, William, Jeffrey Graves, and Lawrence Holder. "Insider threat detection using a graph-based approach." *Journal of Applied Security Research* 6.1 (2010): 32-81.

[15] Spitzner, Lance. "Honeypots: Catching the insider threat." Computer Security Applications Conference, 2003. Proceedings. 19th Annual. IEEE, 2003.

[16] Debar, Hervé, Marc Dacier, and Andreas Wespi. "A revised taxonomy for intrusion-detection systems." *Annales des télécommunications*. Vol. 55. No. 7-8. Springer-Verlag, 2000.

[17] Nguyen, Nam T., Peter L. Reiher, and Geoffrey H. Kuenning. "Detecting Insider Threats by Monitoring System Call Activity." *IAW*. 2003.

[18] Salem, Malek Ben, Shlomo Hershkop, and Salvatore J. Stolfo. "A survey of insider attack detection research." *Insider Attack and Cyber Security*. Springer US, 2008. 69-90.

[19] Magklaras, G. B., and S. M. Furnell. "Insider threat prediction tool: Evaluating the probability of IT misuse." *Computers & Security* 21.1 (2001): 62-73.

[20] Magklaras, G. B., and S. M. Furnell. "A preliminary model of end user sophistication for insider threat prediction in IT systems." *Computers & Security* 24.5 (2005): 371-380.

[21] Kandias, Miltiadis, et al. "An insider threat prediction model." *Trust, Privacy and Security in Digital Business*. Springer Berlin Heidelberg, 2010. 26-37.

[22] Magklaras, G. B., S. M. Furnell, and Phillip J. Brooke. "Towards an insider threat prediction specification language." *Information management & computer security* 14.4 (2006): 361-381.

[23] Teh, Pin Shen, Andrew Beng Jin Teoh, and Shigang Yue. "A survey of keystroke dynamics biometrics." *The Scientific World Journal* 2013 (2013).

[24] Wang, Xuan, Fangxia Guo, and Jian-feng Ma. "User authentication via keystroke dynamics based on difference subspace and slope correlation degree." *Digital Signal Processing* 22.5 (2012): 707-712.

[25] Karnan, Marcus, and N. Krishnaraj. "Bio password—keystroke dynamic approach to secure mobile devices." *Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on*. IEEE, 2010.

[26] Lee, Hyoung-joo, and Sungzoon Cho. "Retraining a keystroke dynamics-based authenticator with impostor patterns." *Computers & Security* 26.4 (2007): 300-310.

[27] Bours, Patrick. "Continuous keystroke dynamics: A different perspective towards biometric evaluation." *Information Security Technical Report* 17.1 (2012): 36-43.

[28] Janakiraman Rajkumar, and Terence Sim. "Keystroke dynamics in a general setting." *Advances in Biometrics*. Springer Berlin Heidelberg, 2007. 584-593.

[29] http://nlp.stanford.edu/IR-book/html/htmledition/support-vector-machines-the-linearly-separable-case-1.html.

[30] Bordes, Antoine, et al. "Fast kernel classifiers with online and active learning." *The Journal of Machine Learning Research* 6 (2005): 1579-1619.

[31] Chang, Chih-Chung, and Chih-Jen Lin. "LIBSVM: a library for support vector machines." *ACM Transactions on Intelligent Systems and Technology (TIST)* 2.3 (2011): 27.

[32] Collobert, Ronan, and Samy Bengio. "SVMTorch: Support vector machines for large-scale regression problems." *The Journal of Machine Learning Research* 1 (2001): 143-160.

[33] V. N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

[34] http://www.ratatype.com/learn/average-typing-speed/

Autonomic Characterization of Workloads using Workload Fingerprinting

Rahul Khanna*, Mrittika Ganguli*, Ananth S Narayan*, Abhiram R† and Piyush Gupta‡
*Intel Corporation
†R.V College of Engineering, Bangalore, India
‡National Institute of Technology Karnataka, Surathkal, India

ABSTRACT

In a cloud service management environment, service level agreements (SLA) define the expectation of quality (Quality-of-Service) for managing performance loss in a given service-hosting environment comprising of a pool of compute resources. Typically, complexity of resource inter-dependencies in a server system often results to sub-optimal behaviors leading to performance loss. A well behaved model can anticipate the demand patterns and proactively react to the dynamic stresses in a timely and well optimized manner. Dynamic characterization methods can synthesize self-correcting workload fingerprint code-book that facilitates phase prediction to achieve continuous tuning through proactive workload-allocation and load-balancing. In this paper we introduce the methodology that facilitates the coordinated tuning of the system resources through phase-assisted dynamic characterization. We describe the method to develop a multi-variate phase model by learning and classifying the run-time behavior of workloads. We demonstrate the workload phase forecasting method using phase extraction using a combination of machine learning approach. Results show the new model is about 98% accurate in phase identification and 97.15% accurate in forecasting the compute demands.

I. INTRODUCTION

In a data center, the workloads utilize server hardware resources to varying degrees of demand. From an end-user perspective, there is a need to provide consistent performance which can be measured in industry known metrics and with standard or other popular benchmarks. Modern server hardware, right from the microprocessor to network controllers, is an increasingly complex system providing various features that are configurable to benefit the target usage scenarios. Modifying one control attribute in isolation can adversely affect the state of other resources which makes tuning a complex optimization problem. Resource tuning requires a proactive analysis based on insights derived from statistical analysis of data collected while running a collection of well-known benchmarks. Proactive self-tuning by advanced knowledge of future states and application behavior enables the system to adapt to changing environments autonomously [1]. Understanding application behavior has been a key source of insight for striking a balance between reducing operational cost while improving the workload performance. Profiling an application reveals its time-varying behavior that repeats in some defined patterns over its lifetime. A program phase defines a discontinuity in time where observable characteristics fluctuates in a distinctive manner to trigger a measurable system impact. Data mining techniques when employed to the runtime trace of a program, can discover the program operational phases and its corresponding properties. The workload characteristics integrated through the sequence of known (or dynamically identified) phase are represented as a fingerprint. These fingerprints ascertain future behavior at relatively low computational cost. Besides the SPEC benchmarks, workloads like Unixbench and specialized workloads by usage are often used to measure the performance of a virtual infrastructure instance allocated to a CPU. Benchmarks developed by end-users measure compute and IO performance to build out the virtual infrastructure for the end-user application. In the public *Infrastructure-As-A-Service* (IAAS) environments the resources are provided in the form of *X* number of cores. A cloud end-user executes applications on these workloads which may either predominantly utilize memory and data storage, or utilize the servers processors. A consistent Quality-Of-Service needs to be maintained for static workload pattern to avoid performance degradation. Without any prior knowledge regarding workload behavioral patterns, sizing of the infrastructure becomes an essential ingredient. In a dynamically changing environment an elastic approach to scaling up or down based on performance requirements and guarantees need to be factored in. This necessitates the need to identify and categorize the workload behavior for static patterns (like batch-mode applications) or dynamic patterns (like interactive applications [2]) to accomplish performance guarantees of the applications operating on workloads. Today, service providers lack processes to commit to a performance SLA that is constant across platform generations, sites, power and thermal variations. The user has limited means to stipulate a unit of required compute performance that is measurable and portable. The usages prevalent in the IAAS are:

1. Application Owner end-user: The application owner or end-user defines the following for each virtual machine (VM):

- Reference design to VM Image
- VM instance metadata
- Type from service catalog to choose a service flavor
- Implied SLA defined as measurable Service-level-Objectives (SLO)
- CSP location
- Credentials

2. The cloud service provider

- A orchestration framework like Openstack [3]

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

- A monitoring and billing framework
- A service management environment to define SLO attached to each cloud instance type
- A policy based remediation framework

In this paper we formulate the need for a workload fingerprint in an *Infrastructure-As-A-Service* (IAAS) management framework. The *Service-Level-Objectives* (SLO) enable a performance metric called service compute Unit (SCU) that corresponds to an active workload in a virtual machine.

$$SCU = fn(F, T, C, I) \tag{1}$$

where,

F = Frequency, T = Throughput, C = cache Efficiency, I = Instruction Set Efficiency.

It is measured, calculated and used as the basis for resource allocation and any remediation action required to guarantee the workload performance. CPU performance telemetry extends the data points for synthesizing the model representation of a workload's fingerprint. A fingerprint pattern facilitates cloud orchestration and service framework to proactively evaluate remedial actions through dynamic resource evaluation and sharing to comply with the service-level-objectives (SLO).

II. Related Work

Exploiting predicted outcomes spatially and temporally for speculative tuning to achieve significant advantage is a not a new concept - it has been applied in several systems of various forms. More specifically, the analysis of programs both offline and online formed the basis of optimizing performance without needing to sacrifice efficiency, in fields spanning from micro-architecture to compute clusters. Gabbay [4] proposes value prediction approach to predict values of operations to achieve instruction level parallelism, which speeds up serial programs. Value and address prediction techniques derived in [5] and [6] help achieve up to 28% reduction in load latency. Warren et al [7] employ genetic search techniques to predict application run time for improving scheduling efficiency and resource utilization. Rahul et al [8] demonstrate device a multi-objective optimization model to optimize performance/watt of enterprise servers by predicting energy with 99.3% accuracy, and [9] where a running average of utilization was used to dynamically configure and limit memory power states in Intelx86 server processors.

Identification and detection of phases and its characteristics has been one of the fundamental techniques of exploiting program behavior. Program phases can be determined completely offline - for example by analyzing Basic Block Distribution Analysis (BBDA) [10] gathered from traces of profiling tools. The frequency of micro-architectural events read from hardware counters forms the basis of several researches, including ours, as they are very fine-grained and incur less system overhead. E. Duesterwald et al [11] argue the need for a predictive system rather than a reactive one, and propose a cross-metric predictor that predicts program behavior with 69% higher accuracy than statistical approaches. In [12] A. Sembrant et al classify phase detection techniques, and propose Scarphase - an online heuristic phase detection algorithm that incurs less than 2% system overhead. T. Sherwood et al [13] provide a comprehensive analysis on phase detection and employ Markov models to predict phases at 80% accuracy for dynamic processor cache size and address width adaptation. Finally, J. Lau et al [14] improve upon phase classification and prediction employing phase confidence metrics along with accurate phase change and length predictions. Workload characterization [15] is a related area of interest where run-time behavior models classify the workloads into well-defined types - memory bound, Input / Output (I/O) bound, cache sensitive, etc. E. Ould-Ahmed-Vall et al [16] characterize SPEC 2006 and SPEC OMP2001 test suites using linear and tree models to study the similarities between them. It is worth mentioning that our work can also be adapted for workload characterization by extracting required information from the corresponding fingerprints.

Prediction and forecasting for cloud orchestration has also been addressed by Saripalli et al [17] and Antonescu et. al. [18] for two distinct usages. Saripalli et. al. uses load prediction to peak its usage (or hotspots) to automatically provision VMs on demand. On the other hand, SLA driven management and efficient use of hardware resources for scaling the application and modifying SLAs (with user intervention) necessitated the use of correlation of SLA metrics and prediction mechanisms by Antonescu. The usage model addressed by Antonescu resembles some of the SLO driven workload management applications in IAAS framework. Antonescu uses periodic time series analysis corresponding to the SLA monitoring metrics and determines runtime on-demand statistical correlations to be used for improving the existing SLAs. His method schedules SLA for different services (like a compute service storage service) for distributed applications. He does not use workload performance (or usage) metrics to derive patterns or phases and correlate its impact on SLA metrics. Saripalli uses ARMA, ARIMA and exponential moving average (EMA) to characterize workload usage patterns from RAM, CPU and disk metrics in a time series fashion in a SAAS environment to elastically load balance.

Eddy et al [19] addresses on-demand resources prediction in the grid and cloud systems to improve autoscaling of resource avocation but does not address scheduling of resources to meet performance targets which necessitated our experiment. In [20] Antonescu et al come close to addressing a problem of Quality of Experience (QoE) of the users and use a combination of forecast-based performance degradation preventive actions and pattern detection. He uses Winter-Holts triple exponential forecasting, autoregressive integrated moving average(ARIMA) and exponential smoothing state space model with Box-Cox transformation, ARMA errors, Trend and Seasonal components (BATS).

Our work differs from the existing work where we perform a workload/phase detection as an integrated approach. This approach allows a uniform definition of phases represented by a three dimensional vector: (a) Workload Type (b) Current Phase identifier and (c) Subsequent Phase identifier. This 3-D structure enables us to perform resource evaluation for shared resources using Naive-Bayesian analysis. For example, evaluating joint probability of a constrained resource demand for independent workload candidates allows us decide if a can-

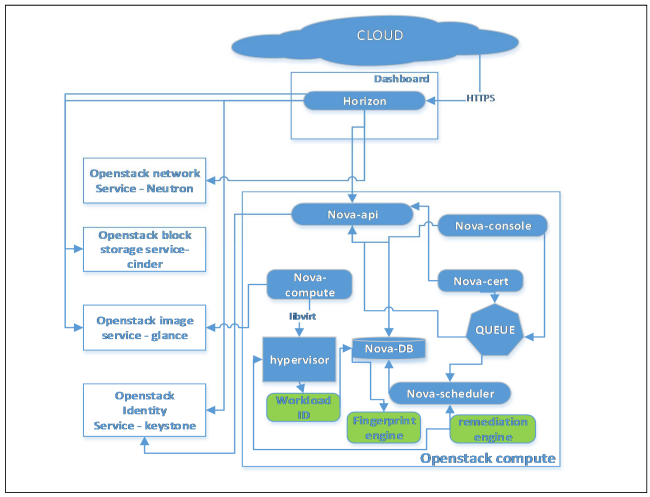


Fig. 1. Open-Stack implementation of Workload Fingerprinting

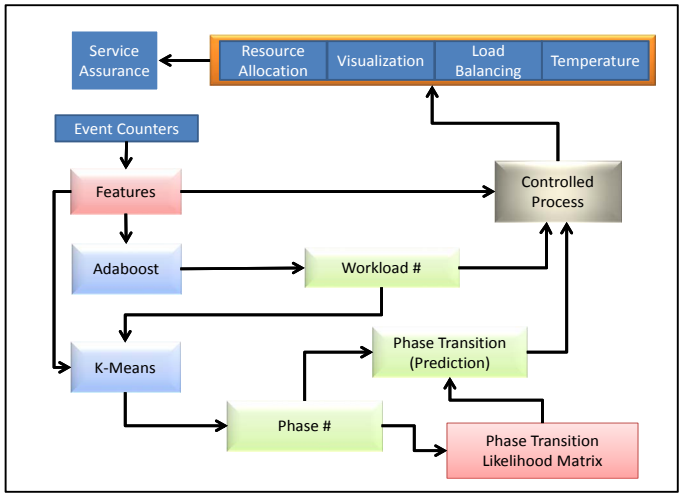


Fig. 2. Two-process phase detection flow. Once workload is identified, phase attributes of workload are calculated. These attributes facilitate statistical evaluation of phase sequences for predictive analysis.

didate VM fits the characteristic of an optimal host container.

III. Background

Workloads are application with specialized objectives (Queries, Searches, and Analysis etc.) undergo phases of execution while operating under multiple constraints. These constraints are related to resource consumption, heat generation and Quality-Of-Service (QoS) requirements. Optimal system operation involves complex choices due to a variety of degrees of freedom for resource and performance tuning. The process involves modeling methodology, implementation choices, and dynamic tuning. Fingerprinting acts as an essential ingredient that captures time-varying behavior of dynamically adaptable systems. This ability is used as statistical output that aids in reconfiguring hardware and software ahead of variation in demand and enables reuse of trained models for recurring phases. Workload Fingerprinting is a method by which individual performance characteristics are collected at a given interval, classified and aggregated to create a fingerprint of the existing workload (or a collection of them).

Figure 1 illustrates an Openstack implementation of hosting the workload fingerprint infrastructure. Openstack is an open source cloud orchestration framework. It builds a list of compatible hosts, i.e. hosts able to run the workload image given the capacity and system characteristics. Openstack Nova-scheduler sorts hosts by fitness which is a metric for prioritizing hosts, assign virtual machines to hosts and starts the workload. A service management agent at the compute host enables monitoring of allocated VMs and workloads to detect performance bottlenecks, and SLO violations. The deep platform telemetry in Intel Xeon based hosts, is collected at each compute host, correlated for usage and performance to create usage and violation scores which in turn are used for service SLO remediation and load-balancing. We add to this framework by adding a ”workload ID” module which enables the Adaboost based workload identification engine, described in Section III-B to generate an ID for the workload running in the VM. We further enhance the infrastructure by adding a ”fingerprint engine” which codes in the k-means assisted solution described in Section III-C. Both the Workload-ID and Phase-ID is saved in the nova-DB or a service management database. In addition, this data is used to make policy decisions assisted by the remediation engine. Some of those could be: (1) migrate out a VM if multiple VMs on the same socket have similar phases occurring at the same time, creating a bottleneck affecting the performance of one or more VMs (2) assign more resources to a VM which is operating at a low performance threshold (3) rearrange the VMs on the different compute resources of a single host to establish a tetris pattern of fingerprints on concurrently run workloads.

A. Forecasting

The workload-Phase forecasting module detects trends in the workload and make predictions about future workload attributes. If the target workload demonstrates a a strong periodic behavior, historic forecast can be incorporated to workload forecasting. This would allow the policy decision to proactively react to the workload spikes ahead of time. This also helps us to take advantage of heterogeneous compute and I/O resources offered by Cloud Computing providers. The workload fingerprinting methodology involves a two-step process that can be summarized as:

1) Workload Detection - Identifying the type of workload
2) Phase Detection - Identifying distinct phases for a given type of workload.

The two-process methodology (Figure 2) allows optimal placement of workloads and proactive provisioning of the resources by forecasting the likelihood of resource demands. The likelihood of phase transitions act as a feedback for resource reconfiguration based on the knowledge of phase specific resource utilization.

B. Workload Determination

Workload identification is a critical step in the process of forecasting the resource utilization at any given time. It represents an abstraction of the actual work that have attributes

that can help in identifying its optimal placement. Workload execution is typically characterized by many different features ranging from cache behavior, CPU utilization, memory bandwidth, I/O bandwidth and so on. The detection process uses a supervised classification process to train a model that detects the type of workload using reduced feature-sets. The learning process infers the underlying relationship between the observed features and a target "workload-type" variable which is subject to prediction. The learning task uses the labeled training data to synthesize the model function that attempts to generalize the underlying relationship between feature vector (input) and workload-type (output). The feature vectors are reduced set of potential candidate features that are extracted corresponding to their significance in the detection process. Less significant or redundant features are discarded to avoid information overhead and computational efficiency of the algorithm.

We use Adaboost [21] algorithm to synthesize workload-detection model.Adaboost is an ensemble method used for constructing a strong classifier as liner combinations of simple weak classifier (or rule of thumb). Similar to any ensemble methods, it employs multiple learners to solve a problem with better generalization ability and accurate prediction. The strong classifier can be evaluated as a linear combination of weak classifiers as below:

$$H(x) = \sum_{t=1}^{T} \beta_t \cdot h_t(x) \tag{2}$$

Where,
$H(x)$= Strong Classifier
$h_t(x)$= Weak or Basis Classifier or a feature

AdaBoost algorithm is adaptive as it uses multiple iterations to produce a strong learner which is well correlated with the true classifier. It iterates by adding weak learners that are slightly correlated with the true classifier. As part of an adaptation process, the weighting vector adjusts itself to improve upon misclassification in previous rounds. The resulting classifier has a greater accuracy than the weak learners classifiers. In general, Adaboost is fast, simple to implement and flexible as it can be combined with any classifier.

Upon completing the training of strong classifier, we evaluate the weak classiers and the significance of the corresponding features. This aids in the selection of the features based on the ranking of weak classiers and their features.

C. Phase Determination

As discussed earlier, workloads are application with specialized objectives (Queries, Searches, and Analysis etc.) undergo phases of execution while operating under multiple constraints. Optimal system operation involves complex choices due to a variety of degrees of freedom for power and performance parameter tuning. Phase detection in a workload acts as an essential ingredient that captures time-varying behavior of dynamically adaptable systems. This ability aids in anticipating and reconfiguring process variables ahead of variation in demand and enabling the reuse of trained models for recurring phases. Phase identification also aids in predicting the future phases during the execution of workloads which prevents any reactive response to a changes in workload behavior. In this context, a phase is a stage of execution in which a workload demonstrates similar power, temperature, resource utilization and performance characteristics.

Dynamical systems are characterized by temporal features whose time-varying properties undergo changes during the operational period. These systems produce temporal sequence of observations that can be analyzed for dynamic characteristics. Training block performs unsupervised classification and builds data structures by partitioning it into homogeneous clusters such that similar objects are classified within the same class. We use K-means clustering algorithm that partitions the d-dimensional features into k clusters such that each emission belongs to the cluster with the nearest mean. k-means is one of the simplest algorithm which uses unsupervised learning method to solve known clustering problems. It works well with large datasets. Its linear time complexity makes it a better choice as compared to other clustering techniques like hierarchical methods. For a given a set of observations $(x_1, x_2, \cdots, x_n)$, K-means clustering algorithm partitions the observations into k sets $S = (S_1, S_2, S_3, \cdots, S_k)$:

$$\underbrace{arg\ min}_{S} \sum_{i=1}^{k} \sum_{x_j \in S_i} \|x_j - \mu_i\|^2 \tag{3}$$

Training block marks the feature components with corresponding cluster attributes that act as database reference for building k Gaussian mixture components $\lambda = (\lambda_1, \lambda_2, \cdots, \lambda_k)$. For a given sequence of d-dimensional observation vector sequence $x = (x_1, x_2, \cdots, x_T)$, the a posteriori probability for i-th mixture component is given by:

$$p(i|x_t, \lambda) = \frac{c_i \cdot G(\lambda_i)(x_t)}{\sum_{k=1}^{K} c_k \cdot G(\lambda_k)(x_t)} \tag{4}$$

where:

$$G_{(\lambda_i)}(x) = \frac{1}{\sqrt{2\pi}^d \sqrt{|\Sigma|}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)} \tag{5a}$$

$$\lambda_i = (\mu_, \Sigma_i) \tag{5b}$$

$$\sum_{k=1}^{K} c_k = 1; \tag{5c}$$

As the features are evaluated for sequence of of d-dimensional observations, these are tagged for cluster assignments which form the basis for resource prediction. Features primarily comprises of elements of utilization, performance, resource consumption and state attributes. In a dynamic resource forecasting, the features are tagged with cluster assignment and processed through the trained model. The workload is fully represented by an N-phase model. These phases represent a synthetic feature that can capture workload's time-varying behavior and can be coupled to dynamic resource variations through an intelligent learning mechanism. These phases can not only define the workload characteristics, but also have a probabilistic transitional behavior which can be applied for long term prediction.

Event Counter	Significance (%)
FREERUN_CORE_ENERGY_STATUS	51.677
OFFCORE_REQUESTS.ALL_DATA_RD	23.775
LONGEST_LAT_CACHE.MISS	7.226
CYCLE_ACTIVITY.CYCLES_L1D_PENDING	6.227
UOPS_ISSUED.CORE_STALL_CYCLES	3.881
SCU	2.655
UOPS_RETIRED.STALL_CYCLES	2.606
CPU_CLK_UNHALTED.THREAD	0.962

TABLE I. SINGLE WORKLOAD CASE: RELATIVE SIGNIFICANCE OF OBSERVED EVENTS FOR WORKLOAD IDENTIFICATION.

Event Counter	Significance (%)
SCU	20.007
STATIC_PKG_THERMAL_STATUS	15.514
UNC_H_REQUESTS.READS_ALL	13.915
UNC_H_IMC_WRITES.ALL	11.747
OFFCORE_REQUESTS.ALL_DATA_RD	10.006
CYCLE_ACTIVITY.CYCLES_L1D_PENDING	9.058
UNC_H_REQUESTS.READS	3.598
LONGEST_LAT_CACHE.REFERENCE	3.589
UOPS_RETIRED.STALL_CYCLES	2.768
FREERUN_PKG_ENERGY_STATUS	2.508
LONGEST_LAT_CACHE.MISS	1.859
CYCLE_ACTIVITY.STALL_CYCLES_L1D_PENDING	1.675
UOPS_ISSUED.CORE_STALL_CYCLES	1.067
FREERUN_CORE_ENERGY_STATUS	0.958

TABLE II. PARALLEL WORKLOAD CASE: RELATIVE SIGNIFICANCE OF OBSERVED EVENTS FOR MULTIPLE WORKLOAD IDENTIFICATION.

IV. EXPERIMENTAL METHODOLOGY

Modern x86 processors typically support a performance monitoring unit (PMU) which can be programmed to collect a variety of hardware performance events. The PMU is exposed through control registers into which the event code is programmed along with additional parameters such as sub event masks, and event counting is initiated. The current event count is exposed through a data register which can be read by software.

Processors typically support a large number of events, but each processor core has a limited set of model specific registers (MSRs) for the PMU. In Intel processors, each core supports 4 programmable MSRs, therefore up to 4 non-conflicting events can be counted simultaneously. For our experiments, we selected four benchmarks from the SPEC CPU 2006 suite - a compute intensive application (POVRay), a memory intensive application (Omnet++), and 2 streaming applications (Milc and LBM) were selected. The hardware performance data was collected using an internal tool. The benchmarks were executed on a dual socket Intel Xeon E5-2680 processor with Intel Hyper-Threading disabled.

A. Workload Identification

Each benchmark was run for three iterations and hardware performance events collected. The benchmarks were pinned to prevent migration between cores. 25 events were selected from the list of all performance events supported by the processor. Only 4 programmable registers are available, therefore, the benchmark execution was repeated in order to collect data for all events of interest. The results from all the rounds of data collection were then parsed and merged to provide a single input set which contained all the events for three iterations of the benchmark. The second experiment involved two benchmarks running simultaneously. Because the benchmarks did not have the same execution time, the shorter benchmark was repeatedly executed till the longer benchmark completed three iterations. Similar to the one benchmark case described earlier, the benchmark execution was repeated in order to collect all events and then parsed to get data for the two benchmarks. Prior to passing the data for learning, it was normalized by dividing the individual event values by the maximum value for the event.

1) Software Architecture for Workload Detection

In this paper workloads are classified based on feature characteristics. We use R platform/language to develop workload classifiers. Since ADABOOST in its native form a binary classifier, we use an improved version called ADABOOST.M1. The boosting function in R allows us to specify a set of dependent and independent variables. In our case, the workload_id is the only explicitly independent variable and hence we create a trained model based on this parameter. Adaboost uses the bagging() function that performs the training over an input dataset for 100 iterations. Over these 100 iterations, the relative significance of the metrics is weighted on the basis of which of the metrics was used how much in the course of the classification.

Once the boosting function has synthesized a trained model, future data can be tested and classified against it using the predict.boosting() function (Listing 1). This function evaluates the new feature data-set using the existing model and assigns it to one of the classes according to the output of the trained model. The experimental results of classification can be verified using the confusion matrix that is returned as one of the output objects of the predict function.

```
1  \\R-code
2    datasample.adaboost<-boosting(wl_id ~.,data=datasample[trainwl, ], mfinal=100 ,
           control=rpart.control(maxdepth=30))
3    b<-datasample.adaboost
4    sortedb<-sort(bwl$importance, decreasing=TRUE )
5
6  Input: Labeled data with values for metrics
7    train := sample(data)
8    //To select a subset of the data to train
9    boosting-op := boosting( independent_var ~., data_to_train, mfinal, size_of_trees)
10   print boosting-op$importance
11   //To show relative importance of variables
12   boosting-op.pred := predict.boosting(boosting-op, data_to_test)
13   //To print confusion matrix
14   print boosting-op.pred$confusion
```

Listing 1. R Program to model workload classifier using Adaboost.M1

B. Phase Detection

Phase identification involves detecting distinctive phases on an identified workload (Section V-A). We perform k-medioids clustering that uses reduced set of feature vectors to identify number of phases present in a workload. Medioids are representative objects of a data set or a cluster with a data set whose average dissimilarity to all the objects in the cluster is minimal.

1) Software Architecture for Phase detection

For Phase-Identification method we use *pamk()* clustering function of *fpc package* [22] of R scripting language. This function is similar to k-means but it operates on medioids. Input parameters for this function are normalized metrics as in Table I and range of the number of clusters. This calls the function *pam()* or *clara()* to perform a partitioning around medioids clustering with the number of clusters estimated by

optimum average silhouette width or Calinski-Harabasz index [23]. The *Duda-Hart test* [24] is applied to decide whether there should be more than one cluster (unless 1 is excluded as number of clusters or data are dissimilarities). By this the function automatically chose the number of clusters present.

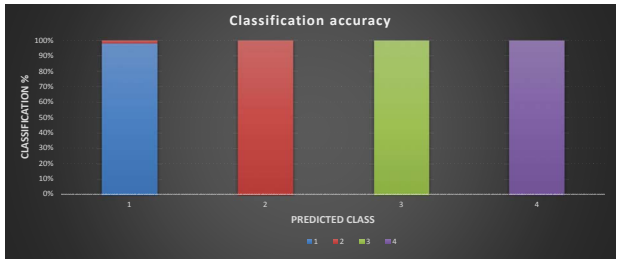
V. Results

A. Workload Identification

The first experiment for workload-identification consisted of executing benchmarks from the SPEC CPU 2006 suite independently on the host. The benchmarks were executed separately, each for four iterations and hardware performance data collected simultaneously. The twenty five events were passed to boosting algorithm in order to train the model. From the data-set, 66% was used as the training set and 33% as the testing set; the training and testing sets were generated by random selection without replacement. The trained model provides us with two outputs: first, identify the most significant events in identifying a workload, and second, a model to classify the workload based on the hardware events. The bagging algorithm identified nine events with non-zero importance in workload classification. The events and their significance are listed in Table I. The model was tested using the test set and the resultant confusion matrix is illustrated in Figure 3. There is a misclassification of 2 samples for one workload (Class 1), and all other workloads are properly classified. The figure above represents the confusion matrix graphically. The individual bars represent the predicted class and the coloring in each bar represents the observed class.

The second experiment for workload-identification involved running two benchmarks simultaneously. The four benchmarks mentioned earlier were executed pairwise on separate cores but on the same processor, therefore they shared resources such as the last level cache on the processor. Cases where two instances of the same benchmark had to run simultaneously were not executed. Similar to the one workload scenario, the dataset was first normalized and then divided into training and testing with 66% a 33% samples respectively, chosen by random selection without replacement. Because two

(a)



(b)

	Observed Class			
Predicted Class	1	2	3	4
1	98	0	0	0
2	2	100	0	0
3	0	0	100	0
4	0	0	0	100

Fig. 3. (a) Classification accuracy of 4-workload, where each workload executes independently at separate instant (b) Confusion matrix

Observed Class

Predicted Class	1-2	1-3	1-4	2-1	2-3	2-4	3-1	3-2	3-4	4-1	4-2	4-3
1-2	686	0	0	0	0	0	0	0	0	0	0	0
1-3	0	696	0	0	0	0	0	0	0	2	0	0
1-4	14	4	694	0	0	0	0	0	0	8	0	0
2-1	0	0	0	694	0	0	0	0	0	0	0	0
2-3	0	0	0	2	700	0	0	0	0	0	0	0
2-4	0	0	0	0	0	700	0	0	0	0	0	0
3-1	0	0	0	0	0	0	700	0	0	0	0	0
3-2	0	0	0	0	0	0	0	700	2	0	0	0
3-4	0	0	0	0	0	0	0	0	698	0	0	0
4-1	0	0	0	0	0	0	0	0	0	690	0	8
4-2	0	0	0	0	0	0	0	0	0	0	700	0
4-3	0	0	0	0	0	0	0	0	0	0	0	692

Fig. 4. Confusion matrix for mixed-workload classification, where each workload executes simultaneously at same instant on a separate virtual machine (VM). Note: 3-4 would mean that workload-3 executes on VM1 and workload-4 on VM2

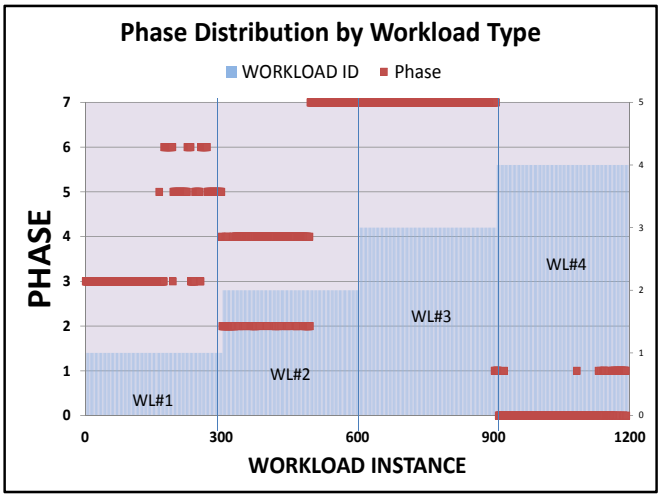


Fig. 5. Workload and Phase dependency graph. Each Workload may share the phase characteristics with other workloads. Additionally each workload can cater to one or more phases.

applications execute simultaneously, the classification must identify both. Given a pair of benchmark, one of the two benchmark is termed the primary and the other termed the interfering benchmark and the label for this pair was obtained by concatenating the label of the secondary benchmark to that of the primary application. For example, with two benchmarks A, and B executing simultaneously, case 1 considered A to be the primary benchmark, and in the second case, B is considered the primary. The workloads during executing interfere with each other, therefore, swapping the primary and interfering benchmark is also valid. Similar to experiment 1, the relative significance of the individual events in predicting the classification of both applications are listed in Table II. The confusion matrix is illustrated in Figure 4. In the matrix, a label A-B and B-A represent the same two benchmarks executing concurrently in the first case A is considered the primary and in the second case, B is considered the primary. The two labels are intentionally considered separately. The confusion matrix

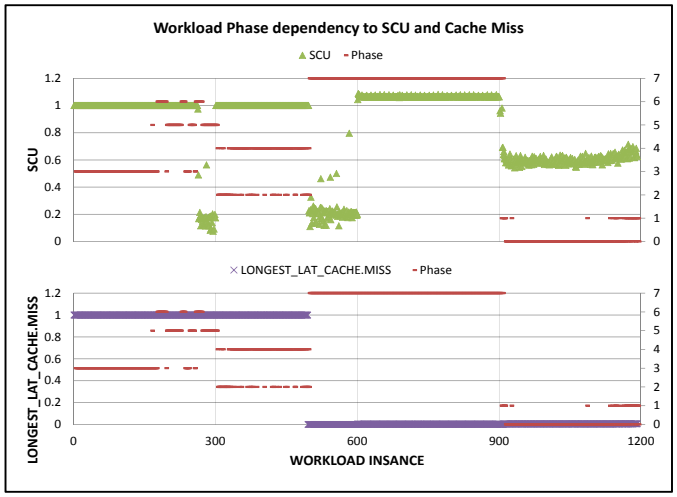
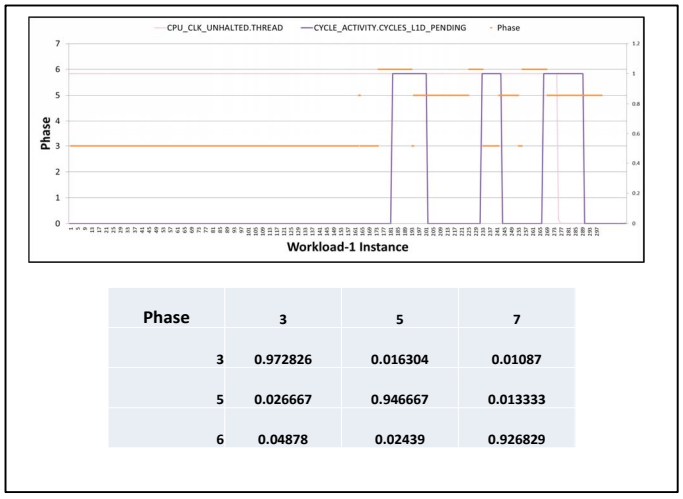


Fig. 6. Workload and Phase dependency graph. Workload-Phase fingerprint shares a unique feature characteristic.



Phase	3	5	7
3	0.972826	0.016304	0.01087
5	0.026667	0.946667	0.013333
6	0.04878	0.02439	0.926829

Fig. 7. Phase transition likelihood matrix and Workload-Phase dependency matrix for workload-1. Phase variations illustrate the behavior of two features (CYCLE_ACTIVITY.CYCLES_L1D_PENDING and LONGEST_LAT_CACHE.MISS

shows a high accuracy of classification of both workloads, with the highest misclassification error being 0.02% in the case of label 1-2. Across all classes the misclassification is .003%.

B. Phase Identification

Phase identification plays a significant role in estimating and reconfiguring the resource utilization for a given set of workloads executing on different Virtual machines (VM). Each phase has a distinctive characteristics of its own and transitions to another phase with certain likelihood. Once we understand the phase transition behavior, we can predict the sequence of phases according to the time spent in each phase. Furthermore, workload-phase relationship facilitates the mixing of diverse workloads among virtual machines by using statistical methods. Figure 5 illustrates the workload-phase boundaries. Each workload is characterized by a unique composition of workload phases called as fingerprints. Once the workload is identified, phase characterize the resource utilization and

Phase #

Phase #	0	1	2	3	4	5	6	7
0	0.943775	0.056225	0	0	0	0	0	0
1	0.384615	0.589744	0	0	0	0	0	0.025641
2	0	0	0.5	0	0.482143	0	0	0.017857
3	0	0	0	0.972826	0	0.016304	0.01087	0
4	0	0	0.201439	0	0.798561	0	0	0
5	0	0	0	0.026667	0.013333	0.946667	0.013333	0
6	0	0	0	0.04878	0	0.02439	0.926829	0
7	0	0.004902	0	0	0	0	0	0.995098

Workload #

Workload #	0	1	2	3	4	5	6	7
1	0	0	0	0.613333	0	0.25	0.136667	0
2	0	0	0.186667	0	0.463333	0	0	0.35
3	0	0	0	0	0	0	0	1
4	0.856164	0.133562	0	0	0	0	0	0.010274

Fig. 8. Phase transition likelihood matrix and Workload-Phase dependency matrix. The Workload-1 phase dependency is highlighted.

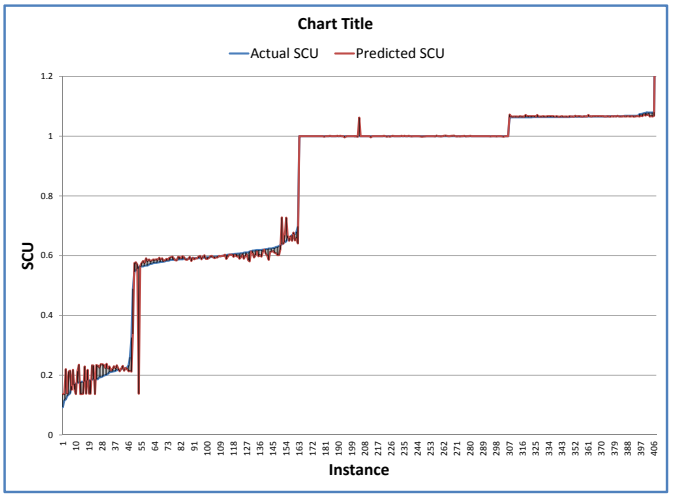


Fig. 9. SCU Prediction using Workload/Phase information feature

its time-series pattern. While workload-3 comprises mainly of phase-7, workload-1 is a complex mix of phases 3,5 and 6. These phases discover the unique characteristic that is specific to a workload instance at any time. Figure 6 illustrates the demarcation boundaries for feature attributes of SCU and cache Miss. For example, while workload-2/Phase-7 corresponds to low SCU(0.2), workload-3/Phase-7 has a high SCU value (1.07). In this case workload bounds the behavior of phase-7. Similarly, in case of workload-2, phase boundaries demarcate the behavior of cache miss. While phases 4 and 2 correspond to a high cache miss (1.0), phase-7 relates to a low value (0).

Figure 7 illustrates the behavior of L1D pending cycles for workload-1. Three dominant phases (3,5,6) capture the dynamic behavior of this feature. Once we identify the instance to be that of workload-1, we may use the phase transition matrix to predict the feature behavior of L1D pending cycles. Although phase-3 corresponds to the low value in the beginning, but when we execute the phase transition from phase-5 to phase-3, the same feature demonstrates a high value.

Figure 9 illustrates the relative accuracy of SCU prediction using Workload/Phase tuple. We evaluate the mean absolute error of 0.99% and relative absolute error of 2.85%. The predictive mechanism allows to regulate the SCU boundaries with a high degree of accuracy. Furthermore, it helps in anticipating the upcoming demands as well as evaluating the quality of mixing the workloads in a server with shared compute resources.

VI. Summary

Optimal resource utilization in a server improves the quality-of-service that leads to the SLO compliance in a data-center. Dynamical systems undergo multiple phases of execution where phase boundaries exist as an optimal choice for initiating or evaluating resource configuration. Dynamic phase detection acts as an essential ingredient for dynamically adaptable systems. Our work presented in this paper is a step forward in achieving a dynamic self-tuning system in a data-center. This helps a class of long running interactive workloads, where phase sequences repeat over longer periods in time. In this paper we developed a workload/phase detection model that use machine learning methodology to classify the workload behavior into various phases of workload operation. The first part of this methodology comprises of workload detection that uses weak-classifiers as in Adaboost. Once the workloads are identified, second part of the methodology uses variants of k-means clustering to synthesize distinctive phases (or behaviors) within workloads. The Workload-Phase feature acts as an additional vector in addition to existing features that improves training models used for SLO related control processes (Load Allocation, Re-Balancing, Resource-Provisioning, Contention-Avoidance). Our results demonstrate a high degree of accuracy (99%) in workload identification and 97.15% accuracy in resource forecasting. Our future work involves auto-synthesis of knowledge-base that maps the fingerprint data to global resource utilization and demands.

References

[1] Rahul Khanna, Mohan Kumar.(2011). Book: A Vision for Platform Autonomy, Robust frameworks for Systems, Intel Press, 2012

[2] Mulia WD, Sehgal N, Sohoni S, Acken JM, Stanberry C L, Fritz DJ. Cloud Workload Characterization. IETE Tech Rev [serial online] 2013 [cited 2014 Jan 6];30:382-97.

[3] Openstack architecture, http://docs.openstack.org/training-guides/content/module001-ch004-openstack-architecture.html

[4] Freddy Gabbay. (1996). EE Department TR 1080, Technion - Israel Institute of Technology, November 1996.

[5] Glenn Reinman and Brad Calder. (1998). Predictive techniques for aggressive load speculation. In 31st International Symposium on Microarchitecture, 1998.

[6] Kai Wang and Manoj Franklin. (1997). Highly accurate data value prediction using hybrid predictors. In 30th Annual International Symposium on Microarchitecture, December 1997.

[7] Warren Smith and Ian Foster and Valerie Taylor. (1998). Predicting Application Run Times Using Historical Information. In Proceedings of the Workshop on Job Scheduling Strategies for Parallel Processing IPPS/SPDP '98

[8] Rahul Khanna, Kshitij Doshi, Christian Le, John Ping, Martin Dimitrov, Mariette Awad and Melissa Stockman, (2011). Dynamic Energy Allocation for Coordinated Optimization in Enterprise Workloads. In Proceedings of the International Conference on Energy Aware Computing, 2011.

[9] Howard David, Eugene Gorbatov, Ulf R. Hanebutte, Rahul Khanna, and Christian Le. (2010). RAPL: memory power estimation and capping. In Proceedings of the 16th ACM/IEEE international symposium on Low power electronics and design (ISLPED '10), 2010.

[10] Timothy Sherwood Erez Perelman Brad Calder. (2001). Basic Block Distribution Analysis to Find Periodic Behavior and Simulation Points in Applications. In Proceedings of the International Conference on Parallel Architectures and Compilation Techniques (PACT), 2001

[11] Evelyn Duesterwald, Calin Cascaval and Sandhya Dwarkadas. (2011). Characterizing and Predicting Program Behavior and its Variability. In Proceedings of the 12th International Conference on Parallel Architectures and Compilation Techniques, 2011

[12] Andreas Sembrant, David Eklov and Erik Hagersten. (2011). Efficient Software-based Online Phase Classification. In Proceedings of the 2011 IEEE International Symposium on Workload Characterization, 2011

[13] Timothy Sherwood Suleyman Sair Brad Calder. (2003). Phase Tracking and Prediction. In Proceedings of the 30th International Symposium on Computer Architecture (ISCA), June 2003.

[14] Jeremy Lau Stefan Schoenmackers Brad Calder. (2005). Transition Phase Classification and Prediction. In 11th International Symposium on High Performance Computer Architecture 2005

[15] S. Huang and W. Feng. (2009). Energy-Efficient Cluster Computing via Accurate Workload Characterization. In Proceedings of 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, 2009

[16] ElMoustapha Ould-Ahmed-Vall, Kshitij A. Doshi, Charles Yount, James Woodlee. (2008). Characterization of SPEC CPU2006 and SPEC OMP2001: Regression Models and their Transferability. IEEE International Symposium on Performance Analysis of Systems and Software, ISPASS 2008

[17] Prasad Saripalli, GVR Kiran, Ravi Shankar R, Harish Narware and Nitin Bindal (2011). Load Prediction and Hot Spot Detection Models for Autonomic Cloud Computing, Fourth IEEE International Conference on Utility and Cloud Computing, 2011

[18] Alexandru-Florian, Antonescu_z, Torsten Braunz (). Improving Management of Distributed Services Using Correlations and Predictions in SLA-Driven Cloud Computing Systems, SAP (Switzerland) Inc., Althardstrasse 80, 8105 Regensdorf, Switzerland, University of Bern, Communication and Distributed Systems (CDS)

[19] Eddy Caron, Frederic Desprez, Adrian Muresan (). Forecasting for Grid and Cloud Computing On-Demand Resources Based on Pattern Matching, 2nd IEEE International Conference on Cloud Computing Technology and Science

[20] Alexandru-Florian, Antonescu_z, Andre Gomesz, Philip Robinsony, Torsten Braunz (2013). SLA-Driven Predictive Orchestration for Distributed Cloud-Based Mobile Services, IEEE International Conference on Communications 2013: IEEE ICC'13 - 1st International Workshop on Mobile Cloud Networking and Services (MCN)

[21] Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci 55(1):119139

[22] R fpc package, http://cran.r-project.org/web/packages/fpc/index.html

[23] Bernard Desgraupes (2013). Clustering Indices, http://cran.r-project.org/web/packages/clusterCrit/vignettes/clusterCrit.pdf

[24] Duda-Hart test for splitting, http://artax.karlin.mff.cuni.cz/r-help/library/fpc/html/dudahart2.html

Cloudy with a Spot of Opportunity: Analysis of Spot-Priced VMs for Practical Job Scheduling

Vedsar Kushwaha and Yogesh Simmhan
Supercomputer Education and Research Centre
Indian Institute of Science, Bangalore, India 560012
Email: vedsarkushwaha@ssl.serc.iisc.in, simmhan@serc.iisc.in

***Abstract*—Public Clouds offer elastic computing resources on-demand using a pay-as-you-go model. While this has opened up access to computing infrastructure, the costs for accessing Cloud resources can be a barrier to adoption in emerging markets. Spot-priced virtual machines (VMs) are offered at deep discounts for the same compute capability as fixed price on-demand VMs. But they can be reclaimed by the Cloud provider at any time, affecting reliability. This paper characterises the behaviour of spot-priced VM from Amazon Web Service for the Asia-Pacific and US East Regions, and analyses their practical impact on running jobs on spot VMs. Our simulation study using jobs of diverse sizes evaluates the trade-offs between cost savings over fixed price VMs and job resilience. Our results show that in most cases, for the workloads studied, we can achieve an effective bottom-line cost savings of 80% using spot VMs, with over 95% reliability.**

I. Introduction

Cloud computing has made utility computing a reality, offering on-demand computing, storage, platform services and Software as a Service (SaaS), using a pay-as-you-go model. Besides helping enterprises outsource their compute infrastructure, this has also democratised access to computational resources – startups can ramp up their compute usage without up-front cost, and scientists in the long tail of computing can periodically access high end resources, elastically [1]. Businesses in emerging markets have also benefited from the lower total cost of ownership (TCO) and ease of accessibility of globally distributed Cloud infrastructure and services [2].

At the same time, the pricing model of Clouds, specifically of Infrastructure as a Service (IaaS) providers, does not offer any discounts to emerging markets compared to their counterparts in developed nations. In fact, customers using virtual machines (VMs) in data centres present in emerging regions end up paying a higher price for such VMs compared to identical VMs in US or European data centres. For e.g., an m3.large VM with 2 virtual CPUs from Amazon Web Services (AWS) [1] costs US$0.140 in the US East Coast Region and US$0.154 in the European Union Region, while the same VM costs US$0.190 from an Amazon data centre in South America, and US$0.196 from their data centre in Asia-Pacific (Singapore). Such a price difference is due to factors such as cost of electricity, infrastructure, personnel, and taxation. Given that customers in emerging markets may prefer geographically proximate data centres, to ensure lower latency and also for compliance with local privacy laws, the premium pricing in emerging nations is a deterrence to Cloud adoption.

Cloud service providers have tried to minimise the operational cost of their unsold Cloud capacity through spot markets. AWS, which introduced spot VM instances in December 2009, remains the most popular of these spot market Cloud providers [3] though others exist [2]. Spot VMs work in a pseudo-auction model. They are often offered at a deep discount compared to fixed price on-demand VMs, despite offering equivalent performance and being available worldwide. As a result, they can significantly reduce the cost of using Cloud resources in emerging markets. However, with the reduction in price comes an additional risk, one of reliability. Spot Cloud providers like Amazon can reclaim spot instances at any time from the users without warning, causing them to lose unsaved data in the VM and disrupting service availability; nominally, the last partially used spot VM hour is not billed. This reclamation action is a function of the spot price set by Amazon and the price at which the user bids for the VM.

Due to this perceived lack of reliability, and also limited literature on spot VMs, spot markets have not gained the kind of traction that fixed price on-demand VMs have. There is some literature on modeling AWS Spot Prices [4], [5], [6] and even using them for Hadoop and SaaS applications [5], [7]. But none take a practical look at characterising the pricing behaviour of spot VMs, and its impact on reliably running jobs. Our earlier work has studied optimal scheduling of jobs on spot VMs to meet deadlines [8]. As we show in this current paper, using Amazon's spot VMs offers highly favourable trade-offs between cost and reliability, with limited need for sophisticated scheduling algorithms or price models. This makes them an attractive opportunity for emerging markets to leverage.

We make the following contributions in this paper: (1) We provide a cost analysis of Amazon's Spot VMs (§ III), specifically comparing the Asia-Pacific (Singapore) region against the US East Coast region, and also performing a study across time, with data from both 2014 and 2012. (2) Further, we perform a simulation study, using real spot price data, on running jobs of diverse compute requirements on such spot VMs, and analyse their savings–reliability trade-offs using metrics we propose (§ IV). Such a characterisation of spot pricing for jobs helps users in SMEs and emerging markets to effectively leverage its full potential.

II. Related Work

Significant research has gone into optimising the cost of scheduling applications on public Clouds [9], [1], [10]. In [11], the authors attempt to automate the match-making between

[1]Amazon EC2 Pricing, http://aws.amazon.com/ec2/pricing/

[2]ComputeNext Cloud Brokerage https://www.computenext.com/

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

the compute requirements of a job and Cloud resources. They use the availability of different VM sizes, and the elasticity offered in acquiring and releasing resources, to make scheduling decisions that are cost-efficient and meet a job's soft deadline. Researchers have also considered streaming jobs and modeled performance variation on Clouds due to multi-tenancy [12]. Jobs using an intelligent mix of on-demand VMs and reserved VMs have also been investigated [13], the latter costing marginally lesser than on-demand VMs but acquired in bulk for extended periods of time. Despite such studies, these are applicable only to on-demand VMs that can be retained by the user as long as they pay the fixed per-hour price, and does not consider the uncertainity of spot-priced VMs, whose dynamic price modeling by the Cloud provider is often opaque. Our work is directed at understanding the easy use of spot VMs for running jobs, and using simulations to bound the job's reliability while helping reduce its monetary cost.

Researchers have examined Amazon's spot prices [3] with the goal of developing prediction models. An early work [14] attempts to reverse-engineer the pricing algorithm using data from US West region in 2009-2010. They posit that Amazon's spot price does not follow a market-based demand-supply model, but is based on a constantly changing internal reserve price. Other research has gone into using Markov Chain models to capture spot price variations [4], [5]. [6] examines cloud computing pricing dynamics across Amazon EC2 regions to discern the opportunity for arbitrage, and test for the influence of latency as a pricing wedge in the observed pricing dynamics. A detailed statistical analysis of spot prices is provided by [15], and it proposes a Gaussian mixture model to capture the spot pricing. Our work is in a similar vein, in trying to understand the spot price behaviour. However, rather than reflexively trying to predict the changes in pricing or accurately model it, we go on to examine how even a limited understanding of the pricing can translate into effectively running jobs on spot VMs. Our job simulation study estimates the practical impact of spot pricing on a job's reliability, and the savings from using spot over on-demand VMs.

There has been work on developing frameworks and job scheduling algorithms for spot-priced VMs. [16] uses an economics-based approach to develop scheduling policies when there is resource uncertainty. They investigate profit-aware job admission control and scheduling over resources that have an uncertainity in their availability and pricing in the future. [7] examines SaaS running on IaaS spot instances, and how the SaaS provider can charge its customer for executing its services and paying them a penalty for failing to meet service level agreements. They propose a spot VM bidding scheme and VM allocation policy designed to optimise the average revenue earned per time unit. Our own prior work [8] uses check-pointing and migration strategies to increase the reliability of jobs running on spot VMs, using existing price prediction models. Others [5] have also leveraged the robustness of the Hadoop MapReduce framework to mitigate the impact of running on less reliable, but cheaper, spot instances. These research involve non-trivial job analysis, spot prediction models and scheduling strategies to run on spot instances, many of which are not translatable to practice. Here, we instead examine when "good enough" is enough, and show that even simple price analysis offers scheduling insights on spot VMs that offer adequate cost-benefit trade-offs to many applications.

At a more abstract level, research has explored how Cloud pricing models and markets can be developed and used by service IaaS providers and Cloud brokerages [17]. These are intended to help improve resource utilisation, reduce VM pricing, enhance quality of service for customers, and maximise the profit for Cloud service providers [18]. While such literature offers formal models for Cloud vendors, there is little evidence to show that sophisticated pricing and market-based models have been adopted in public Clouds at large scale, partly because the system design and assumptions imposed by such research may not hold in reality, and also because Cloud providers and users often prefer simplicity in practice.

III. Analysis of Spot Pricing

A. Dataset and Virtual Machine Description

Amazon Web Service (AWS) is the largest provider of spot VMs on public Clouds globally, in addition to on-demand VMs offered as part of their IaaS. For our analysis, we consider four different VM types that Amazon recommends for general purpose computing: *m1.small, m1.medium, m1.large,* and *m1.xlarge* [3] [4]. We abbreviate these as **Small, Medium, Large** and **XLarge**, respectively, in the rest of the paper. We also consider two different AWS regions (or data centres), one in Singapore for Asia-Pacific (*ap-southeast-1a*) and the other in Northern Virginia for US East Coast (*us-east-1a*). We abbreviate these as **AP-SE** and **US-E**, respectively. AP-SE is important for geo-location with Asian markets. All four VM sizes are available in both these regions, with identical performance but different on-demand prices (Table I)[5].

TABLE I. Performance and on-Demand Price for VM Types

VM Type	Performance				†On-Demand Price [US Cents]	
	Virtual CPUs (vCPU)	Elastic Compute Units (ECU)	Memory [GiB]	Storage [GB]	AP-SE	US-E
m1.small	1	1	1.7	1 × 160	5.8	4.4
m1.medium	1	2	3.75	1 × 410	11.7	8.7
m1.large	2	4	7.5	2 × 410	23.3	17.5
m1.xlarge	4	8	15	4 × 410	46.7	35.0

† *As on July 15, 2014*

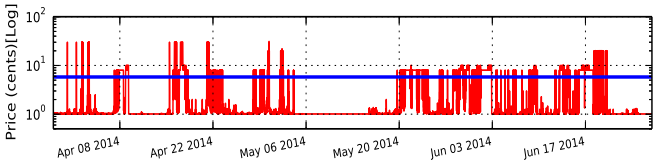
We collect spot price data [3] for two different time periods that are separated by 15 months: *Aug–Dec, 2012* for US-E, and *Apr–Jun, 2014* for AP-SE and US-E, for these four VM types using AWS's Command Line Interface [6]. We abbreviate these time periods as **2012** and **2014**. AWS reports the spot price only when it changes. We discretise this into uniform-spaced spot prices at $1\ min$ intervals for our analysis. If prices change within one minute, we consider the maximum price within that interval; this happens fewer than 25 times in the $1.918\ million$

[3] Amazon EC2 Instances, http://aws.amazon.com/ec2/instance-types/

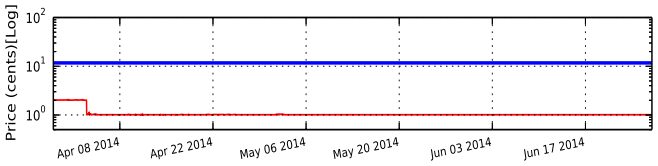
[4] AWS recently introduced *m3.** instance types as an upgrade from *m1.**. We use *m1.** in this paper to allow analysis across time: 2012 and 2014. We expect *m3.** to have similar pricing behaviour, and our results to carry forward.

[5] On-demand VM prices change too, but over the period of months than hours. For simplicity, we use the static on-demand VM prices at the time of writing (July 15, 2014). So the on-demand VM price in 2012 would be different from this. However, the on-demand prices have consistently fallen over time. So the comparison we make between spot VM prices in 2012 with on-demand VM prices in 2014 is actually favourable to on-demand VMs.

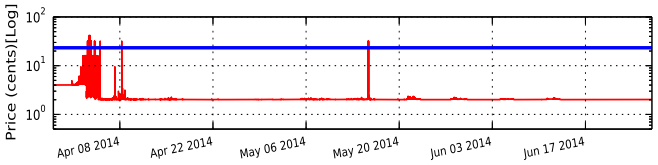
[6] AWS only provides the past 3 months of spot price data. Since the accuracy of non-AWS provided historic data is unknown, we use AWS data that we directly collect: in 2014 for AP-SE, and in 2012 and 2014 for US-E.



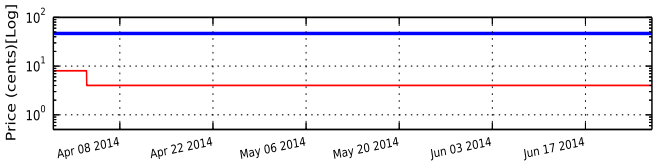
(a) Small VM



(b) Medium VM

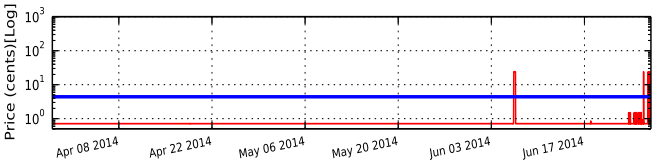


(c) Large VM

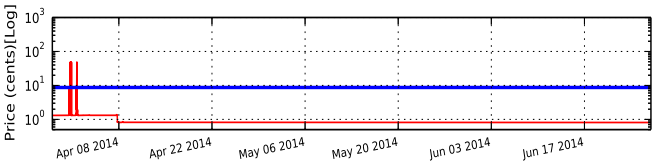


(d) XLarge VM

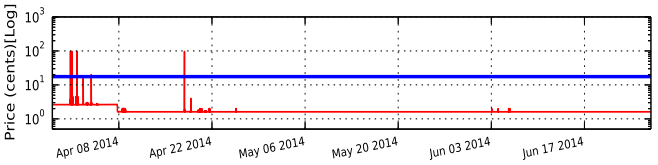
Fig. 1. AWS Spot prices (log scale) observed in *Apr–Jun 2014* in *AP-SE*. For reference, blue horizontal line shows fixed on-demand price for that VM.



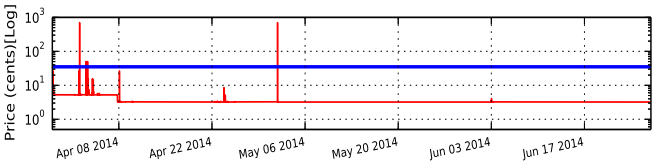
(a) Small VM



(b) Medium VM



(c) Large VM



(d) XLarge VM

Fig. 2. AWS Spot prices (log scale) observed in *Apr–Jun 2014* in *US-E*. For reference, blue horizontal line shows fixed on-demand price for that VM.

spot price intervals we consider overall. Figs. 1, 2, and 3 show the uniform-spaced discretised spot prices for the four VM types in AP-SE 2014, US-E 2014 and US-E 2012 [7].

B. Variation in Spot Prices

Figs. 1, 2 and 3 show the observed spot prices in US Cents (¢) per VM-hour over time. The blue solid line in each plot indicates fixed on-demand price for this VM, also provided in Table I. We observe that the spot price is often below the on-demand price, but there are sharp spikes when the spot price rises to much more than the on-demand price, sometimes peaking to US$10 per VM hour (Fig. 3(d)). These spikes are intermittent, and may indicate AWS trying to flush spot VM users due to internal demand [8].

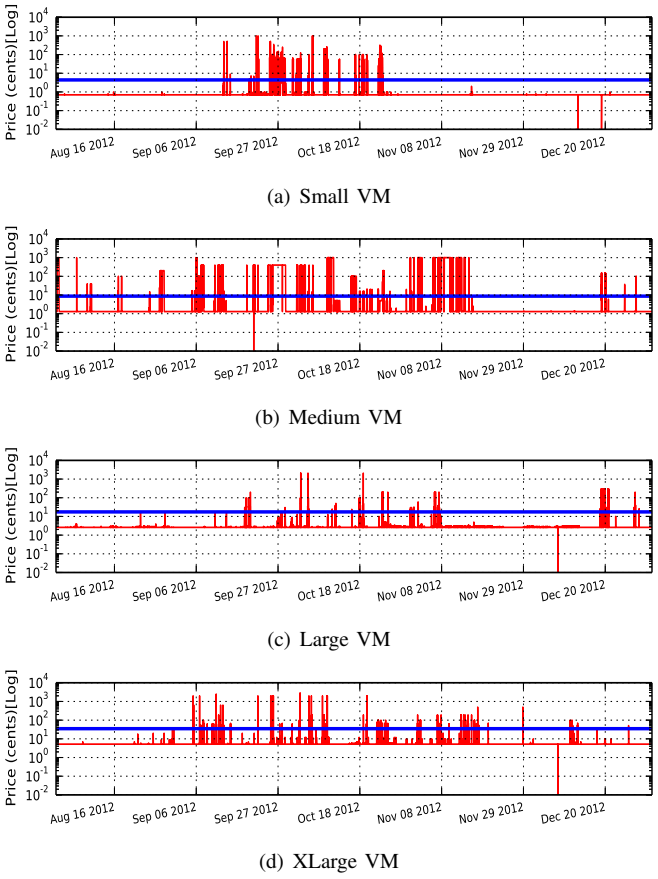
Unlike upward price spikes, we notice that the minimum spot price for each VM type does not go below a threshold value for a region. As shown in Fig. 4, which plots the number of virtual CPUs (vCPUs) per VM along X Axis against its minimum observed spot price, this threshold spot price value is proportional to the number of vCPUs for AP-SE and US-E in 2014. Note that both small and medium VMs have 1 vCPU though they have 1 and 2 ECUs respectively, hence the constant minimum price for Small and Medium. This lower bound may correlate with the operating cost for AWS to run that VM type (e.g. power, cooling, personnel, taxes, etc.) at that data centre. US-E in 2012 (Fig. 3) has a few outliers, where the spot price has dropped briefly to US$0.0001. Hence its minimum observed price appears flat and close to US$0.00.

The prices across regions appear to be uncorrelated. For e.g., the Pearson's correlation coefficient (ρ) between spot prices for AP-SE and US-E in 2014 for each VM type are $\rho_{Small} = -0.023$, $\rho_{Medium} = 0.242$, $\rho_{Large} = 0.052$, $\rho_{XLarge} = 0.093$. While this lack of correlation may provide price arbitrage opportunities across regions [6], the price in the US-E is often smaller than the price at AP-SE. So applications may be better off bidding for instances in US-E if geo-location, network proximity or legal policies are not a constraint. For e.g., in 2014, the spot prices of US-E VMs were smaller than AP-SE VMs during these fraction of times: $\Delta_{Small} = 99.39\%$, $\Delta_{Medium} = 94.34\%$, $\Delta_{Large} = 95.91\%$, and $\Delta_{XLarge} = 94.56\%$. Note that the fixed prices of on-demand VMs in US-E are also cheaper their AP-SE counterparts (Table I).

Interestingly, despite no consistent price trends across regions, we do notice a distinctive step-down pattern in the spot prices for Medium, Large and XLarge VMs in AP-SE and US-E in Apr 2014. The price drops sharply by 50% for all VM types in AP-SE, followed by a similar drop for these VM types in US-E a few days later. This may indicate that a Cloud fabric upgrade or pricing algorithm upgrade is being rolled out in one data centre first, followed by other data centres. For e.g.,

[7] Pricing datasets used in this paper are available at http://dream-lab.serc.iisc.in/data/

[8] Is Amazon effectively killing the spot market for EC2 instances? Sep 2011. https://forums.aws.amazon.com/message.jspa?messageID=281545



(a) Small VM

(b) Medium VM

(c) Large VM

(d) XLarge VM

Fig. 3. AWS Spot prices (log scale) observed in *Aug–Dec 2012* in *US-E*. For reference, blue horizontal line shows fixed on-demand price for that VM.

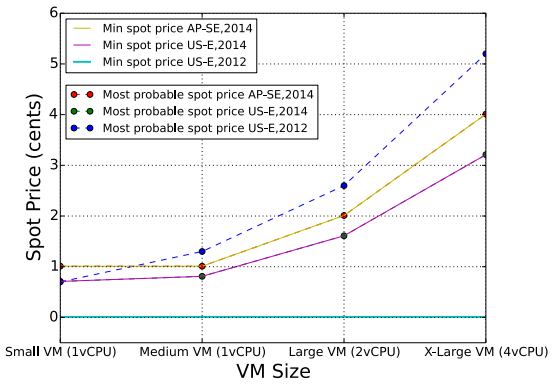


Fig. 4. The *minimum observed spot price* for each VM type, across regions and years, are in solid line. The *most probable spot price*, for the same VM types, region and year, are in dotted line with marker. Note that both these plots coincide exactly, except for US-E 2012.

AWS dropped their on-demand prices [19] on Apr 1, 2014.

C. Spot Price Probability Distribution

The probability density function (PDF) for the discretised spot prices is calculated for the four VM types in AP-SE 2014, and US-E 2014 and 2012. These show the normalised frequency of occurrence of a particular spot price within the time periods considered in 2014 or 2012. For brevity, we only show the plots for medium VMs in Figs. 5(a), 5(b), and 5(c); plots for other VMs sizes are comparable.

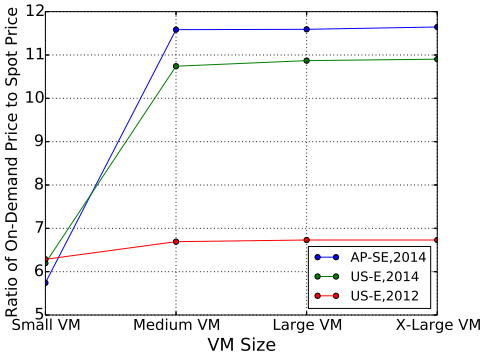


Fig. 6. Ratio of fixed on-demand price to the most probable spot price. The most probable spot price is $5-12\times$ cheaper than the on-demand price.

We notice from Figs. 5(b), and 5(c) that the PDF of spot prices for US-E has changed between 2012 and 2014, with the range of probable values narrowing down from $10^{-2}-10^{+3}$ to $10^{-1}-10^{+2}$. This may be a seasonal characteristic within a year, or a changing price pattern across years.

Interestingly, a single spot price typically appears a majority of the time for a region and VM, during a time period. In fact, such a price occurs $>80\%$ of the time in most cases, except for AP-SE Small and Large in 2014, when it occurs $>40\%$ of the time. Table II shows the most probable price and its frequency for different VMs, regions and periods. This indicates that spot prices often tend toward a particular probable value. In fact, when we overlay the most probable spot price on top of the minimum observed spot price in Fig. 4, these two values coincide (except for US-E 2012 that has a few outliers in minimum observed cost). This suggests that Amazon may often offer spot prices at near operational cost.

TABLE II. MOST PROBABLE SPOT PRICES AND THEIR PROBABILITY

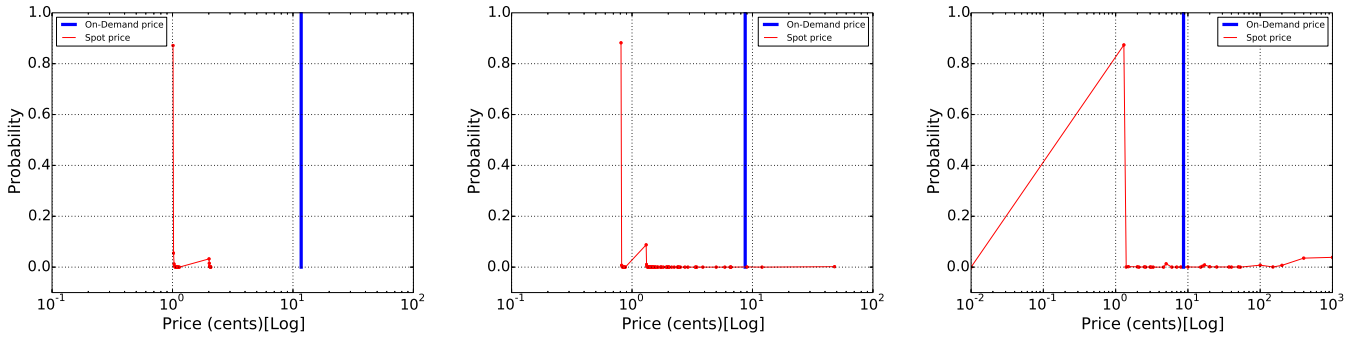
Region	Time Period	VM Type	Spot Price [US Cents]	max(Pr)
AP-SE	2014	Small	1.01	0.41
AP-SE	2014	Medium	1.01	0.87
AP-SE	2014	Large	2.01	0.48
AP-SE	2014	XLarge	4.01	0.94
US-E	2014	Small	0.71	0.99
US-E	2014	Medium	0.81	0.88
US-E	2014	Large	1.61	0.88
US-E	2014	XLarge	3.21	0.80
US-E	2012	Small	0.70	0.95
US-E	2012	Medium	1.30	0.87
US-E	2012	Large	2.60	0.86
US-E	2012	XLarge	5.20	0.96

NOTE: Numbers in red highlight $<80\%$ probability for most probable spot price

When we compare the most probable spot price against the equivalent fixed price on-demand VM in a region in Fig. 6, we notice that the former is at least $5\times$ cheaper than the on-demand price. As we increase the VM size, the price advantage between most the probable spot price and the fixed on-demand price increases to almost $12\times$ for XLarge VM in AP-SE in 2014. Larger spot VMs thus offer an enhanced price benefit.

D. Rate of Change in Spot Prices

To estimate the dynamism of spot price changes, we count the frequency of spot price changes with in a single day, both upward and downward. We also count the magnitude of price decreases and increases each day. Say $\mathcal{D}_j$ is the j^{th} day in a given time period, τ_j^{start} and τ_j^{end} are the timestamp for the start (midnight) and end of day D_j, $\mathcal{S}^{\nu}(t)$ is the spot price



(a) AP-SE, 2014 (b) US-E, 2014 (c) US-E, 2012

Fig. 5. Probability distribution of spot prices (log scale) for *Medium VMs*, in different regions and time periods. For reference, blue vertical line shows fixed on-demand price for medium VM in each region.

at timestamp t for VM type ν. The *frequency of total price changes* in the day $\mathcal{D}_j$ for a VM ν is given by:

$$\mathcal{F}^{\nu}_{tot}(D_j) = \sum_{m=\tau_j^{start}}^{\tau_j^{end}} 1 \mid \mathcal{S}^{\nu}(m) \neq \mathcal{S}^{\nu}(m+1)$$

where m is in minute increments. Further, the frequency of daily price *increases* and *decreases* is calculated by including the conditions $\mathcal{S}^{\nu}(m) < \mathcal{S}^{\nu}(m+1)$ and $\mathcal{S}^{\nu}(m) > \mathcal{S}^{\nu}(m+1)$, respectively. Similarly, the net *magnitude* of price changes, $\mathcal{M}^{\nu}_{tot}(\mathcal{D}_j)$, in a day $\mathcal{D}_j$ is given by:

$$\frac{\sum_{m=\tau_j^{start}}^{\tau_j^{end}} \mathcal{S}^{\nu}(m+1) - \mathcal{S}^{\nu}(m) \mid \mathcal{S}^{\nu}(m) \neq \mathcal{S}^{\nu}(m+1)}{\mathcal{F}^{\nu}_{tot}(D_j)}$$

Likewise defined for the magnitude of price increases and decreases, $\mathcal{M}^{\nu}_{inc}(\mathcal{D}_j)$ and $\mathcal{M}^{\nu}_{dec}(\mathcal{D}_j)$, within a day $\mathcal{D}_j$.

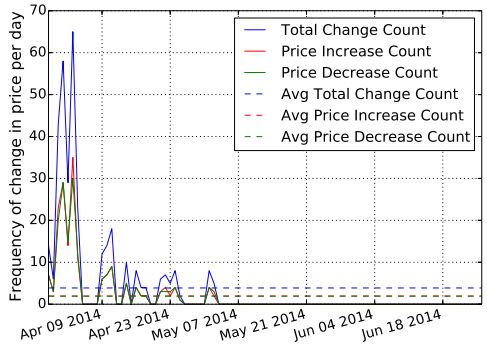
We plot the frequency and magnitude of spot price changes per day for medium VM in AP-SE in 2014 in Fig. 7. For brevity, we omit plots for other VM sizes, regions and time periods, but the results are similar. Surprisingly, despite the price changes appearing to be intermittent in Figs. 1–3, we see that the number of times a price changes in the positive and negative direction within each day is almost identical. While the number of changes per day varies (e.g. Jun, 2014 has no changes but early Apr, 2014 has > 50 changes per day in in Fig. 7(a)), these are symmetric in terms of frequency. Furthermore, the magnitude of positive and negative changes within a day are themselves similar, which means that despite prices changes within a day, the net price change at the end of a day tends to zero, as seen in Fig. 7(b).

Table III shows the correlation between the number and magnitude of spot price increases and decreases within a day, across VMs, regions and time periods. But for a few exceptions in red, we see that $\rho > 0.950$ for both frequency and magnitude, strongly indicating that net change in either direction is conserved within a 24 hr period. In fact, we see a similar conservation (though slightly weaker; not shown) within a 12 hr period too. This suggests that Amazon's spot pricing is incremental/symmetric in nature, and that prices that go up tend to come down, and vice versa, and are highly conserved within a single day. As a result, this periodicity can be exploited in designing spot VM bidding strategies.

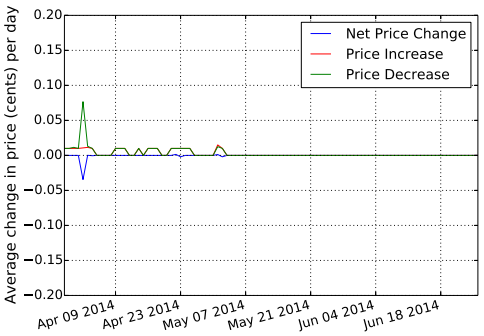
TABLE III. CORRELATION BETWEEN THE FREQUENCY/MAGNITUDE OF PRICES INCREASES AND DECREASES WITHIN A 24 HOUR PERIOD

Region	Time Period	Small	Medium	Large	XLarge
Correlation between Freq. of Increase and Freq. of Decrease					
AP-SE	2014	0.990	0.995	0.998	0.703
US-E	2014	0.981	0.999	0.992	0.991
US-E	2012	0.984	0.985	0.996	0.992
Correlation between Mag. of Increase and Mag. of Decrease					
AP-SE	2014	0.991	0.648	0.999	-0.009
US-E	2014	0.862	0.999	0.995	0.994
US-E	2012	0.973	0.989	0.989	0.976

NOTE: Numbers in red highlight < 0.950 *correlation*



(a) Frequency of Spot Price Changes per day. Horizontal dashed lines show averages across the entire period.



(b) Magnitude of Spot Price Changes Per Day

Fig. 7. Frequency and Magnitude of change in spot prices – increases, decreases and total changes – per day for *Medium VMs* in *AP-SE, 2014*.

IV. ANALYSIS OF JOBS ON SPOT VMS

In this section, we translate our observations on spot prices into their corresponding impact when running jobs on spot VMs through a simulation study. We make several simplifying assumptions that help generalise our job analysis. Individual jobs run exclusively on a spot VM, each with a resource

requirement (*job size*) specified in terms of *ECU core-minutes*. Elastic Compute Unit (ECU) is a normalised unit of compute capability reported for VMs by Amazon. Small, Medium, Large and XLarge VMs of the *m1* class have 1, 2, 4, and 8 ECUs, respectively. We assume these jobs are CPU bound, and can fit in the memory, storage and network capacity available on any of the VM types listed in Table I. We consider diverse job sizes: $10, 30, 60, 240, 480$, and 1440 core-mins. The wall clock duration of each job is defined as $\delta = \frac{Job\ Size}{ECU\ of\ VM}$. E.g., a 30 core-min job will take 15 mins to complete on a Medium VM, while a 1440 core-min job will take 3 hrs on an XLarge VM. Unless a spot VM is reclaimed, both spot and on-demand VMs of the same size take the same duration for a job.

To run jobs, we bid for spot VMs at prices that are fractional values of the fixed on-demand prices of the same VM type in that region. We consider bid prices at 70%, 80%, 90%, 100%, and 110% of the on-demand price. Amazon assigns spot VMs if the bid price is greater than the current spot price, and these VMs are retained only until the bid price remains above the spot price, i.e., if the spot price increases above the bid price, it is an *out-of-bid event*, the VM reclaimed, and the job on it is terminated with all progress lost. For simplicity, we do not change the bid price once defined.

A job is charged only at the spot price, even if the bid price is greater. Billing is in hourly increments, and charges accrue immediately at the VM hour boundary (or partial hour, if the job completes within the hour). The spot price when the VM is acquired is used as the billing rate for the following VM hour. Each subsequent hour uses the current spot price at the start of that VM hour. However, if an out-of-bid event happens at anytime in-between, the partial hour used is not billed. Any whole hours used before the last partial hour, though, is charged even as the job has failed.

We use the uniform-spaced spot prices at $1\ min$ intervals available for the four VM types in AP-SE in 2014, and US-E in 2012 and 2014 for our simulation study. Each job size is "started" on every VM type in each region and time period, at each minute interval. Thus, for a 30 core-min job, we simulate its run $(90\ days \times 24\ hours \times 60\ mins - 29\ mins) = 129,571$ times for a Small VM in AP-SE during Apr-Jun 2014. We do likewise for the 6 job sizes, 4 VM types and 3 regions/periods, for a total of about $9.31\ million$ simulated job runs.

A. Definitions

1) Successful and Failed Jobs: A job is successful if the spot price of the VM on which the job is running remains less than the bid price of that VM, during the entire duration of job. If the spot price of the VM becomes greater than the bid price at any point during the job's duration, the job fails.

2) Reliability: The Reliability (or success rate), $\mathcal{R}$ for a job is defined as:

$$\mathcal{R} = \frac{Number\ of\ Successful\ Jobs}{Total\ Number\ of\ Jobs\ Attempted}$$

The $Failure\ Rate$ for a job is $(1 - \mathcal{R})$.

3) Savings: The relative savings for a Job $\mathcal{J}_\delta$, of wall clock duration δ when running on VM of size ν, started at time τ_i:

$$\mathcal{P}^\nu(\mathcal{J}_\delta, \tau_i) = \sum\nolimits_{h=0}^{\delta} \mathcal{O}^\nu - \mathcal{S}^\nu(\tau_i + h)$$

where h is in hourly increments, $\mathcal{O}^\nu$ is the fixed price of on-demand VM of size ν, and $\mathcal{S}^\nu(t)$ is the spot price of VM of size ν at timestamp t. Note that savings is defined only for successful jobs, and it accumulates by the hour.

The cumulative normalised savings for the above job, started at each minute boundary of the spot price time period $\langle \tau^{begin}, \tau^{end} \rangle$ (e.g. $\langle$1 Apr 2014, 30 Jun 2014 $\rangle$), with a reliability of $\mathcal{R}$ is:

$$\mathcal{P}^\nu\%(\mathcal{J}_\delta, \tau^{begin}, \tau^{end}) = \frac{\sum_{m=\tau^{begin}}^{\tau^{end}-\delta+1} \mathcal{P}^\nu(\mathcal{J}_\delta, m)}{(\mathcal{O}^\nu \times \delta) \times (\mathcal{R} \times n)}$$

where $n = \big((\tau^{end} - \delta + 1) - \tau^{begin})\big)$ is the number of jobs simulated in time period $\langle \tau^{begin}, \tau^{end} \rangle$, and m is in minute increments. $(\mathcal{O}^\nu \times \delta)$ is the job's cost on fixed-price on-demand VMs, and $(\mathcal{R} \times n)$ gives the number of successful jobs.

4) Loss: The relative loss for a Job $\mathcal{J}_\delta$, of wall clock duration δ when running on VM of size ν, started at time τ_i and with an out-of-bid event occurring at the hour $\widehat{\delta}$ from the start time is given by:

$$\mathcal{L}^\nu(\mathcal{J}_\delta, \tau_i, \widehat{\delta}) = \sum\nolimits_{h=0}^{\widehat{\delta}-1} \mathcal{S}^\nu(\tau_i + h)$$

where h is in hourly increments, and $\widehat{\delta} < \delta$. Note that loss is defined only for failed jobs, and it accumulates by the hour.

Similarly, the cumulative normalised loss for the above job, started at each minute boundary of the spot price time period $\langle \tau^{begin}, \tau^{end} \rangle$, with a reliability of $\mathcal{R}$ is:

$$\mathcal{L}^\nu\%(\mathcal{J}_\delta, \tau^{begin}, \tau^{end}) = \frac{\sum_{m=\tau^{begin}}^{\tau^{end}-\delta+1} \mathcal{L}^\nu(\mathcal{J}_\delta, m, \widehat{\delta})}{(\mathcal{O}^\nu \times \delta) \times ((1 - \mathcal{R}) \times n)}$$

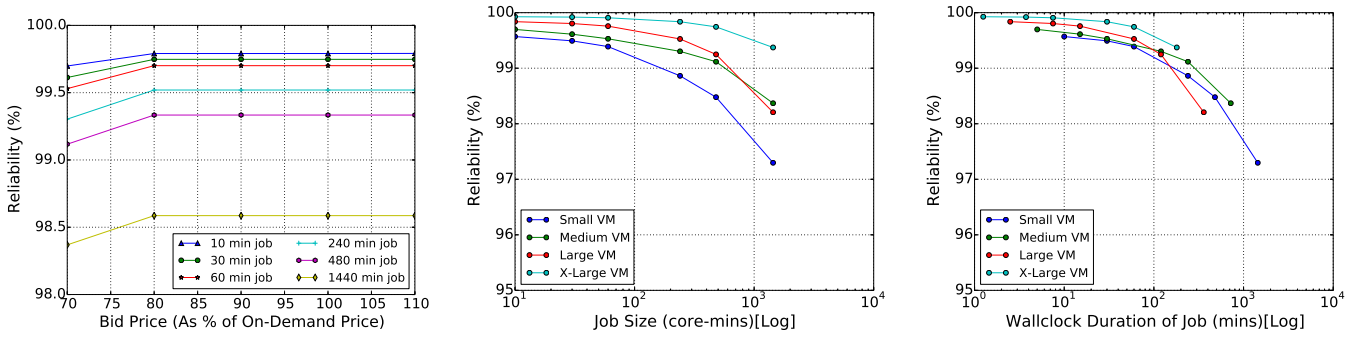
where $\widehat{\delta}$ is a function of the spot price when each job is simulated, and $((1 - \mathcal{R}) \times n)$ gives the number of failed jobs.

5) Effective Savings: The reliability weighted savings, or effective savings, for a job with reliability $\mathcal{R}$, normalised savings $\mathcal{P}\%$ and normalised loss $\mathcal{L}\%$ is given by $\mathcal{R} \times \mathcal{P}\% - (1 - \mathcal{R}) \times \mathcal{L}\%$.

B. Reliability of Jobs

Since spot VMs have a perceived lack of robustness, first we discuss the reliability of jobs on spot VMs before their potential cost benefits. Fig. 8(a) shows a representative plot of how the reliability of various sized jobs change as the bid price increased from 70% to 110% of the on-demand price, for Medium VM in US-E in 2014. Some observations are that *the reliability is relatively high across the board, at above* 98%, and there is barely an improvement of 1% as the bid price goes from 70% to 110%.

We focus on bid prices at 70% of on-demand price, to characterise the lower end of reliability among our parameter space. Fig 8(b) shows the reliability varying with different VM sizes for US-E 2014, as the job size varies along the X Axis, using a 70% bid price. A common trend that we see is that as the job size increases, the reliability decreases. This is understandable, since the larger the job, longer its run duration and greater the chance of an out-of-bid event. AP-SE 2014 and



(a) *US-E 2014, Medium* (b) US-E, 2014 (c) *US-E, 2014* at *70%* of on-demand price

Fig. 8. Reliability of jobs with different parameters run on spot VMs from US-E in 2014. Note that the Y Axis scale is different across plots.

US-E 2012 shows similar results of a decrease in reliability with increase in job size, and are omitted for brevity.

Longer job durations can increase the chance of an out-of-bid event. We derive a stronger correlation between the wallclock time that a job runs for on a VM and the reliability of jobs on that VM. Fig. 8(c) illustrates this for US-E in 2014, as a complement of the Fig. 8(b). Here, the plots are more closely grouped across VM sizes, indicating $\sim 1\%$ drop in reliability for every 200 $mins$ increase in a job's duration.

C. Savings and Loss of Jobs

The normalised savings for a job is the fractional benefit that can be gained from running it successfully on spot VMs, relative to the cost on an equivalent fixed-price on-demand VMs. For jobs that fail due to out-of-bid events, there is a cost paid for whole VM hours used without accomplishing the job. We calculate the normalised loss for a failed job as the loss from running it on spot VMs, relative to running it successfully on a fixed-price on-demand VM. Fig. 9 plots these values when bidding at 70% of on-demand price. Across all regions, *we gain a savings of over* 80% *by using spot VMs* over on-demand VMs. This corresponds to the ratio between most probable spot price and on-demand price being $5\times$ or more in Fig. 6.

Notice that as the job sizes increase, we do not see a tangible change in the savings % gained. This is understandable, since the normalised savings comes from the difference between the spot and on-demand price (§ IV-A3). As spot price does not change often, the gains remain constant. We also see that AP-SE and US-E Small VMs offer a smaller savings, at 80% compared to 90% for the other VMs. From Table II, US-E Small VM's spot price at US$0.0071, in 99% of the time, while Medium is at US$0.0081, in 88% of the time. So a Small spot priced VM with half the compute power of a Medium costs almost as much most of the time. The on-demand prices of a Small is however half as much as an on-demand Medium VM. *As a result, small spot-priced VMs offer lower savings.*

The loss that is suffered on account of failed jobs is also relatively small, and often limited to $< 5\%$, with the only exception being when using Small VMs on US-E in 2014, with large jobs; it rises to 7%. We do see that as the job size increases, the probability of loss also increases linearly. It grows from 0% for jobs that fit within one wallclock hour of a VM, by about 1% for every additional 400 $mins$ of job size. Note that this loss does not include the lost opportunity cost of failing to run the job. While not plotted, we report that there is negligible impact of the bid price increasing from $70\% - 100\%$, but we do see the average loss % grow sharply by $\sim 5\%$ when bidding over the on-demand price at 110%.

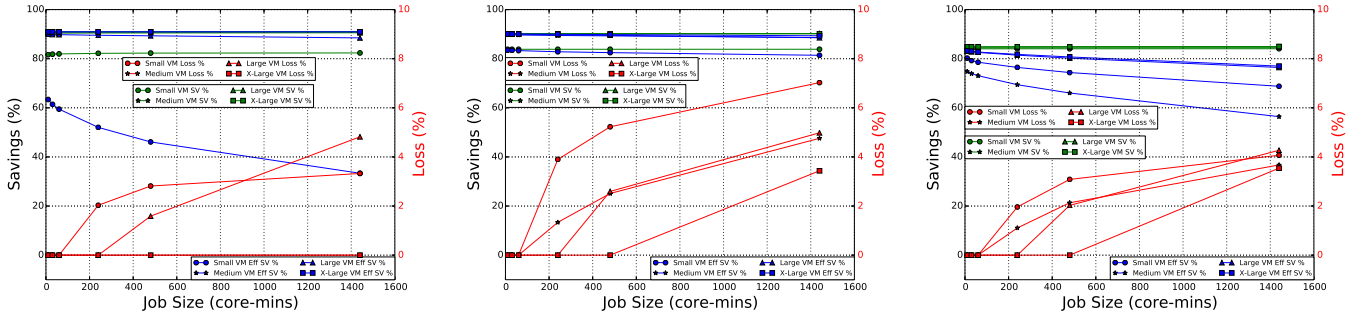
The effective savings offers a reliability weighted function over savings and loss. This is the bottom-line savings (ignoring the lost opportunity cost of failed jobs). Fig. 9 shows these in blue lines. In most cases, *the effective savings is fairly high at* 90%. However, US-E in 2012 shows lower effective savings across VMs as the job size increases, as does AP-SE in 2014 for Small VM. Notice that in Fig. 8(b), the reliability for VMs drops with the job size. So we see the effect of a linear combination of decreasing reliability and increasing normalised loss in Fig. 9. *Note that Small VMs suffer a dual penalty*: jobs run for a longer duration on them, increasing the chance of an out-of-bid event, and the loss also accumulates over more hours for them.

Lastly, we look at a scatter plot of the normalised savings % vs. the reliability % for different VM types, across regions and time periods, in Fig. 10. The colors are grouped by the VM type, so VMs that cluster along the top right of the plot offer significant savings with high reliability, on an average across the jobs. We see that for AP-SE in 2014, all but small VMs consistently offer a $90\%+$ *savings over on-demand with a* $95\%+$ *reliability*. Medium, Large and XLarge are equally good in US-E in 2014, with even the Small offering $> 95\%$ reliability and $> 85\%$ savings. US-E in 2012, however, offers lower reliability for Small and Medium and a slightly diminished savings of 85% across the board.

V. Conclusion

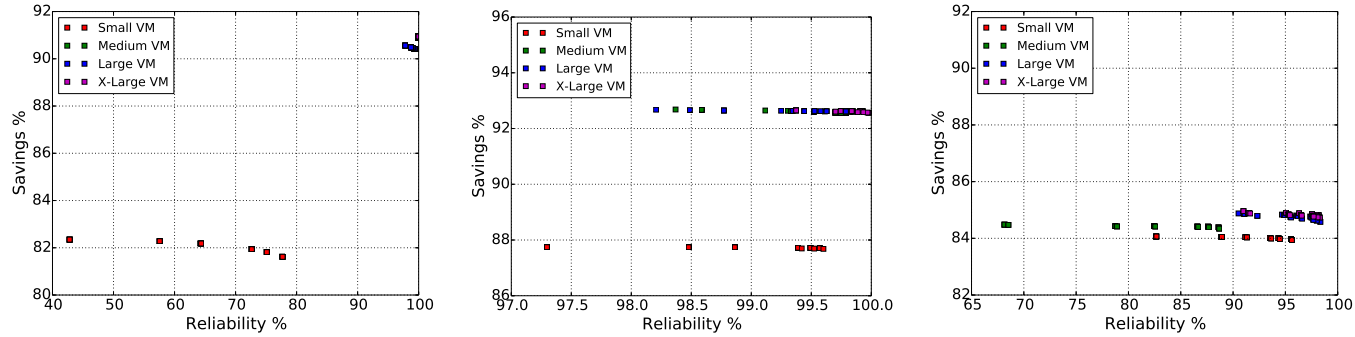
In this paper, we have provided an analysis of AWS spot prices for the AP-SE and US-E regions. We see that prices across regions appear to be uncorrelated except when major pricing changes or software updates happen. There are seasonal and annual variations in the pricing pattern, but the probability that the spot price is at the minimum observed price is fairly high. Also, larger spot VM offer a better price advantage over on-demand VMs. It is interesting to note that the number and magnitude of price changes in the upward and downward directions are conserved within each day.

More so, we have mapped its impact, through a simulation study, on running diverse job sizes. Using meaningful metrics such as reliability and effective savings, our study offers key



(a) AP-SE, 2014 (b) US-E, 2014 (c) US-E, 2012

Fig. 9. Primary Y Axis shows Normalised Savings% (Green) and Effective Savings% (Blue), as Job Size increases on X Axis. Secondary Y Axis shows Normalised Loss% (Red) for the job sizes. We bid at 70% of on-demand price.



(a) AP-SE 2014 (b) US-E 2014 (c) US-E 2012

Fig. 10. Scatter plot of Reliability % vs. Average Savings % for different VMs, across regions and time periods. Each marker represents a job size, and colors are assigned by VM Size. Note that X Axis scales vary across plots.

insights into practically using spot-priced VMs relative to on-demand VMs. The reliability of jobs is relatively high across the board, at above 98%, with barely an improvement as the bid price goes beyond 70% of on-demand VM price. We see effective savings of over 80% in most cases when using spot VMs, with 90% effective savings observed in 2014 data. Small VM offer lower savings due to several factors, and are less preferred. These results suggest that AWS spot instances are highly favourable for cost-conscious enterprises in emerging markets.

REFERENCES

[1] W. Lu, J. Jackson, and R. Barga, "Azureblast: A case study of developing science applications on the cloud," in *HPDC*, 2010.

[2] N. Kshetri, "Cloud computing in developing economies," *Computer*, vol. 43, 2010.

[3] A. W. Service, "Amazon ec2 spot instances," Jul 2014, http://aws.amazon.com/ec2/purchasing-options/spot-instances/.

[4] M. Zafer, Y. Song, and K.-W. Lee, "Optimal Bids for Spot VMs in A Cloud for Deadline Constrained Jobs," in *IEEE CLOUD*, 2012.

[5] N. Chohan, C. Castillo, M. Spreitzer, M. Steinder, A. Tantawi, and C. Krintz, "See spot run: Using spot instances for mapreduce workflows," in *USENIX HotCloud*, 2010, pp. 7–7.

[6] H. K. Cheng, Z. Li, and A. Naranjo, "Cloud computing spot pricing dynamics: Latency and limits to arbitrage," University of Florida, Tech. Rep., 2012.

[7] M. Mazzucco and M. Dumas, "Achieving performance and availability guarantees with spot instances," in *HPCC*, 2011.

[8] H.-Y. Chu and Y. Simmhan, "Cost-efficient and resilient job life-cycle management on hybrid clouds," in *IEEE IPDPS*, 2014.

[9] G. Lee, B.-G. Chun, and H. Katz, "Heterogeneity-aware resource allocation and scheduling in the cloud," in *USENIX HotCloud*, 2011.

[10] D. Warneke and O. Kao, "Exploiting dynamic resource allocation for efficient parallel data processing in the cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 6, 2011.

[11] M. Mao and M. Humphrey, "Auto-scaling to minimize cost and meet application deadlines in cloud workflows," in *Supercomputing*, 2011.

[12] A. G. Kumbhare, Y. Simmhan, and V. K. Prasanna, "Plasticc: Predictive look-ahead scheduling for continuous dataflows on clouds," in *IEEE/ACM CCGrid*, 2014.

[13] Y.-J. Hong, M. Thottethodi, and J. Xue, "Dynamic server provisioning to minimize cost in an iaas cloud," in *SIGMETRICS*, 2011.

[14] O. A. Ben-Yehuda, M. Ben-Yehuda, A. Schuster, and D. Tsafrir, "Deconstructing amazon ec2 spot instance pricing," *ACM Transactions on Economics and Computation*, vol. 1, 2013.

[15] B. Javadi, R. K. Thulasiram, and R. Buyya, "Statistical modeling of spot instance prices in public cloud environments," in *UCC*, vol. 1, 2011.

[16] F. I. Popovici and J. Wilkes, "Profitable services in an uncertain world," in *Supercomputing*, 2005.

[17] F. Teng and F. Magoulès, "A new game theoretical resource allocation algorithm for cloud computing," in *International Conference on Advances in Grid and Pervasive Computing*, 2010.

[18] B. Sharma, R. K. Thulasiram, P. Thulasiraman, S. K. Garg, and R. Buyya, "Pricing cloud compute commodities: A novel financial economic model," in *CCGrid*, 2012.

[19] AWS, "Aws price reduction," Mar 2014, http://aws.amazon.com/blogs/aws/aws-price-reduction-42-ec2-s3-rds-elasticache-and-elastic-mapreduce/.

Contracts in Cloud Computing

Irene Kafeza

National Academy of Legal Studies Research (NALSAR)

Hyderaband, India
kafeza@nalsar.ac.in

Eleanna Kafeza
Business School, Dept. Marketing and Communication
Athens University of Economics and Business Athens, Greece
kafeza@aueb.gr

Epameinondas Panas School of Science and TechnologyDept. of Statistics

Athens University of Economics and Business
panas@aueb.gr

***Abstract*— In an increasingly integrated global economy the importance as well as the growing availability of Cloud Providers has provided companies, individuals and the Governmental agencies with a variety of benefits such as significant cost reduction. As the role and number of Cloud Providers has increased a novel set of issues has emerged. These novel issues refer to a variety of legal complexities from the initial choice of the proper Cloud Provider as well as the appropriate contract for the deployment of its services.**

This paper is concerned with the novel issues arising of the deployment of Cloud Computing contracts. It presents the related issues and discusses ways and suggestions by which the legal framework could be demystified so that the contracts conducted in the Cloud Computing environment can be conducted efficiently and in a legal manner, for the benefit of the private as well as the public sector. It is concerned that the current legal framework cannot provide solution for effective deployment of Cloud Computing contracts and drawing on this evidence, it discusses the steps that Cloud Computing participants need to take to correctly identify their contractual rights and obligations.

Keywords—legal issues; cloud contracting; SLA contracts;

I. INTRODUCTION

Cloud Computing services meet the worldwide business demand for intense problem solving capabilities and makes it possible to share Computing resources on an unprecedented scale among geographically distributed participants. The Cloud Provider allows the dynamic discovery of Computing resources, the immediate allocation and provision of the resources, as well as the management and provision of secure access to those resources. This redefinition of allocation and distribution of services arises as an emerging Computing paradigm and is expected to change the IT landscape including technology, business and services.

Cloud Computing environment refers to a collection of heterogeneous Computing resources that are shared by many individuals and organizations.These resources are collaborating collaborating to offer more effective solutions to a variety of business problems. A successful Cloud business model requires a secure platform that will enable safe and stable collaboration of various resource owners and service users. This requirement is twofold: on one hand a secure technical infrastructure has to be in place and on the other hand a legal framework has to be introduced to increase confidentiality and enable predictability of commercial transactions on the cloud. From the legal point of view several issues have to be addressed in order to create a trusted Cloud environment for business transaction

Cloud Computing has been viewed as an opportunity for faster, better and cheaper services. It has lower costs for storage and computing, is quick and cheap to setup, and allows for flexibility by making applications available at remote offices, on the road, via a smartphone, or from a home PC. It is expected that in the future a typical workplace will include several Cloud Computing applications. An important feature of this service is that all issues regarding the update of software licenses, software updates, hardware failure, adding of capacity are resolved by the SaaS Cloud provider. However, cloud computing deployment raises some concerns such as how can the user trust the Provider that the service will be always available or whether the Provider can be trusted with sensitive data. Private and public sector organizations when entering into a Cloud Provider agreement for rendering its services must be sure that the Cloud Computing function is in compliance with the legislation. Most importantly the user needs to know his rights and obligations as well as the Cloud provider's rights and obligations towards him.

The existing Cloud computing environment lacks a legal framework that allows safe transactions among the organizations that dynamically form the cloud. At the same time, the way contracts can be created and managed in a Cloud Computing environment is of paramount importance for planning and implementing clouds.

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

II. CLOUD COMPUTING CONTRACTS

A. Cloud computing non-negotiated contracts

1) Background

The core issue in the adoption of Cloud Computing services is related to the adoption of the proper contractual model for the Cloud services. Selecting the right Cloud service Provider is the first critical step towards the Cloud Computing adoption. Depending on the user's needs the Saas or Pass or IaaS type of Cloud Computing will be selected. The choice of the appropriate service is vital for the specific purpose for which users employ the Cloud environment. After the selection of the right service, the user needs to enter into a contract with the selected Cloud service provider. This contract can be either a negotiable contract or a standard form contract.

There is different terminology used for Cloud computing contracts. Other companies use the term "Master agreement" for the main agreement with the cloud user and "Service Level Agreement "for the separate contract that regulates the specification of the services. Other companies use the term "Terms of Use" for the general contract and other companies use the "Term of Use "for the general contract that incorporates the Service Level Agreement. For the purpose of this work the terminology "terms of use" or "Cloud Provider contract" means the main contract which incorporates the Service Level Agreement. Therefore, in this work, the Cloud Provider Contract refers to the general clauses of the contract with the Service Provider as well as includes the Service Level Agreements (SLA) which refers to the terms for the specification of services e.g. scalability etc.

There is not a uniformly accepted definition of what constitutes contract. Contract is a promise or a set of promises for the breach of which the law gives a remedy or the performance of which the law in some way recognizes as a duty [1]. An agreement enforceable by law is a contract [2].

An electronic contract extends these definitions to electronic agreements; agreements created through electronic means. These agreements can be classified either as negotiable agreements where the parties can negotiate the terms of the contract or standard form agreements in which the contract is drafted by one party and the other party can only "take it or leave it". The standard form agreements (or adhesion contracts or boilerplate contracts or mass market contracts or Terms of Service) appeared when the computer software industry was faced with huge loss of money due to piracy of software. Although the reproduction of software without authorization is prohibited by the Copyright legislation, the IT companies could not efficiently prohibit the unauthorized reproduction.

The main reason was that the enforcement of rights of the copyright owner under the copyright law has been proved difficult and inefficient. Thus, the software companies attempted to enforce their rights through standard form contracts that restricted the user's rights. These contracts initially were the shrink wrap contracts, (where the license is included in a software package that covers the purchased software and can be read only after the buyer of the software opens the plastic wrap of the software), the click wrap contracts(electronic form agreements set up by one party to which the other party may assent by clicking on the "I agree" button) and browse-wrap contracts (binding agreements simply by visiting the site without the need of clicking any button).

2) Cloud computing contracts as click-up contracts

One of the above categories of standard form contracts, the click wrap contracts, is the usual type of contracts that the Cloud Provider and the Cloud user enter into. These contracts are referred to as Cloud provider's contracts or Service Level Agreements. The Cloud Provider either provide the client user with one document to sign that is titled Master Agreement or Service Level Agreement or Terms of Service and incorporates terms regulating the quality of its service 's performance or the Cloud Provider gives two set of contracts to the user to sign. Thus, the first contract refers to the general terms of service (Master Agreement) and the second contract (Service Level Agreement) refers to the quality of Cloud Provider Services.

The Cloud Provider contract incorporating the Service Level Agreement is usually presented on the Cloud Provider's site and the user has to click the acceptance ("I agree") button in order to proceed to the Cloud Provider Contract. These types of contracts are convenient and cost effective for the Cloud Providers since they don't need to bargain individually with each user therefore saving costs for personnel salaries, time etc. However, the main drawback of these types of contracts that have been challenged on the Courts is that they do not comply with the notion of entering into valid contract according to the principles of traditional contract law. Particularly, their enforceability has been challenged on the ground of the inequality of power between the parties since the vendor is the one who drafts the terms in his favour. The Courts in order to come with a justifiable solution to these issues have applied the "assent analysis "test in order to determine the validity of such contract. The "assent analysis" is to examine whether the user has actually clicked the acceptance icon or has proceeded in a manner that would be impossible to proceed without clicking the acceptance button. Thus, if the Court is satisfied that the user has actually accepted the click wrap contract by clicking the "I agree" button, holds the click wrap contract enforceable. The Courts have denied the validity of click wrap contracts only in cases in which the user has not been adequately informed that his assent is needed or his assent was not required in order to proceed to the contract thus, he has not clearly accepted.

For example, in the Microsoft Azure site in order to proceed to Azure you need to click the button that states " I Agree to the Windows Azure Agreement , Offer Details and Privacy Statement" otherwise you cannot sign in. This is a click wrap contract and the user is bound from its term once he clicks" I agree". But to what exactly he is bound? He is bound to the Windows Azure Agreement. To enter this agreement the user needs to click the button, which is in an obvious position on the site, thus adequate notice is there that qualifies as informed consent that leads to a binding contract.

This agreement includes a term that states "Microsoft Azure Agreement consists of the below terms and conditions as well as the SLAs". In order to read the SLAs the user needs to go to the left of the website and click the relevant link that takes them to the SLAs. The question that arises is: is the SLA included in the click wrap contract? According to the Courts decisions, someone may argue that is not since the user needs to go through all these buttons to enter into the Service Level Agreement. This process does not qualify as" informed consent" although the agreement is on the site because the user has to go all these clicks in order to read the SLA. Thus, the user is bound only by the first click wrap agreement which is presented in an obvious manner and request for a clear acceptance while the SLA is not part of the click wrap contract since it is not presented in a clear and obvious manner to the user. The fact that the click wrap agreement mentions that the SLA is included in the contract does not constitute adequate notice which is required for the valid acceptance. Therefore, the Azure user has a contract for the general terms but not a binding contract for the SLA.

3) User's perspective

Moreover, the contracts with the Cloud Service Provider (including the Service Level Agreement) are often long and include terminology that is not easily understood by the user. Sometimes, the terms are not provided in a visible and easy accessible way to the users. Some other times, the users don't have the time to read them. The result is that Cloud users are not sufficiently informed about their rights and obligations when entering into a contract with a Cloud provider. The general principles of contract law apply in these cases where, as discussed above, the Cloud Providers should have clear terms, ask for the affirmative acceptance of the Cloud user as well as the Terms of Use and Service Level Agreement must be presented in an obvious manner to the user before he enters into the contract. Otherwise, the validity of contract is questionable since the assent of the user might not qualify as adequate acceptance resulting in a binding contract. On the other hand, if the Cloud Provider has clear terms, provides adequate notice to the Cloud user and the user accepts, a valid contract is formed.

4) Limitation of Liability and Disclaimers

Although the contract with the Cloud Provider might be valid, it still might contain certain terms that their validity is questionable. Courts have invalidated specific terms on click wrap contracts when the specific terms are extremely onerous for the other party. One set of terms that have been discussed and created controversies are the terms regarding limitations of liability and disclaimers referring to the services rendered.

One of these terms is the limitation of liability clauses. Liability of the Cloud Provider might arise where the Cloud user has suffered damage because he relied to the information provided by the Cloud Provider which was inaccurate or false. These limitations of liability clauses give the right to the Cloud Provider to disclaim its liability usually through disclaimers. These terms exclude liability of the Cloud Provider for non-performance of its services or false performance and a variety of other instances in which the services might not be provided as prescribed in the contract. Considering that he primary function of Cloud Provider is the assurance of the quality of the service, the question that naturally arises is what is the use of using the service since the Provider that provides it and its primary responsibility is to assure its accuracy has disclaimed all responsibility.

The issue of liability refers to a variety of instances where the things might go wrong. Can an exception clause exclude liability for these instances and will the courts accept such clause as valid? Since there is no specific legislation that addresses these issues a question arises whether the Cloud Provider could by itself pose the standards and rules upon which its conduct will be based and state in its Service Level Agreement what obligations and responsibilities is willing to undertake. Moreover, it is doubtful whether the Cloud Provider will be liable only for losses caused from reliance on erroneously performed services or additionally for no-self compliance with its policies. The liability issues associated with these questions are related with the degree of fault and the extent to which the Cloud Provider is able to disclaim or limit his liability. The question is: what is the point of entering into a contract with a Cloud Provider to employ its services if it is not certain whether it is going to provide the requested service and there is not any consequence for this? The Cloud Providers have argued that the provision of service is cheap and thus one cannot expect to have high standard of assurance while paying so low. Cloud Providers argue that if we try to force Providers to accept more liability while asking them to maintain low commodity prices, the result will be Providers to undermine market development [3].

Absent a coherent legal framework the Cloud Provider could by itself pose the standards and rules upon which its conduct will be based and state in its Terms of Use what obligations and responsibilities is willing to undertake. In this case the wording of the contract will determine the contractual obligations of the parties. Therefore, to ensure enforceability the parties should focus on the following issues: Notice and consent: have the parties clearly and explicitly given their informed consent to conduct the transaction? Have adequate notices provided by the Cloud Provider? Have signature formalities required for this transaction been satisfied? Are copies of the Cloud Provider and user contract available to all parties?

Although the limitation of liability clause is an important one, there are a set of other clauses that the Cloud user might not agree upon. The Cloud Contract usually includes a term that the Provider can change the terms without any notification. The user in this case, either might attempt to negotiate the change of this term and include a term that the Cloud Provider should inform him about the changes as well as a term that he has the right to terminate the contract if he does not agree with new terms.

.

5) Liability under Tort Law

Cloud users would be entitled to recover damages against a Cloud Provider for a breach of contract based on reasonable reliance on performance of its services as stated in its Service Level Agreements Additionally, the Cloud user could base his claims against the Cloud Provider on tort law for negligence if he can demonstrate that the Cloud Provider didn't show the care he ought to show for the provision of its services. Since the core business of the Cloud Provider is the provision of professional level of specific services it seems that he should comply with higher standards of professional care This kind of professional liability for negligence apply to persons that are professionals having specialized knowledge and skills so other people put special trust on them due to this specialization, thus the courts apply to them a higher standard of care than of that of a reasonable man. It seems that the Cloud Provider's fall under the definitions of professional duty of care. Nevertheless, it is questionable whether this is a correct approach and whether higher negligence standards would be more suitable to the nature of their businesses to be imposed to them.

Although we employ the traditional doctrines of contract law to deal with these situations still there is uncertainty regarding the obligations and liabilities of Cloud Providers which self- limiting almost any liability of theirs while their role is to provide assurance and trust for the service. The existing legislation is not addressing particularly these issues thus it seems that either a new legislation should be enacted or the burden of solving this complex situation will be on the courts.

B. Cloud Computing negotiated contracts

In other instances the contracts with the Cloud Providers are negotiable and not presented in a standard form. These contracts –incorporating Service Level Agreements or titled as Service Level Agreements) should specify what type of Cloud service will be provided to the customer to ensure that: a) key elements required for Cloud services (warranties, guarantees, performance metrics, etc.) are not left out of the SLA and therefore rendered unenforceable, b) common terms and definitions are used within the SLAs to avoid costly misunderstandings between parties, and c) to create an environment which allows agencies to objectively compare competing services [4]. These contracts can be signed either with the physical presence of the parties or through electronic means. The" Functional Equivalence" approach or "non-discrimination" doctrine which does not deny enforceability of a contract solely on the ground that it is in electronic form is almost universally accepted principle. Thus, a contract with a Cloud Provider might be entered into electronically and it is valid as long as the substantive law requirements are fulfilled including the signing of the parties.

Most common is the conclusion of these contracts through email. AS long as the writing and signing requirements are fulfilled, these are valid contracts. When negotiating these contracts it might be better if parties include clauses clarifying the form of possible later amendment. There is a discussion whether an oral modification clause or email exchanges could amend a contract thus it is advisable to clarify the form of later modifications of these contracts.

1) Challenging the identity of specifications

Moreover, the Cloud user should read carefully all the terms in the Cloud Provider contract. Sometimes, there is a term for free services. Users should be skeptical about the availability of "free" services since there is another term that charges for the "free" services. There are terms that allow the transfer of the user's data in third parties. Users should clarify with the Cloud Provider the meaning of third parties and decide whether they would like to agree on such a term. Sometimes there is term that in busy working hours there would be a downtime. Depending on the user's needs this might be a term that nullifies the usability of the contract.

The renewal term is also important. The term should be clear whether the service would be renewable automatically and what happens if no. For example in case that the term in not renewable what happens with the Cloud users data: will be kept by the Cloud Provider or it will return them to the user.

The Cloud user must check whether there is a term that allows subcontracting of the Cloud Providers services or a term that allows the transfer of control to another Cloud Provider. In these cases, the Cloud user should negotiate to enter a term that allows him to read the terms of the subcontracting entity and give him the right to terminate the contract in case he disagrees with the sub-contracting. The Cloud user should also include a term in the Service Level Agreement that if the Cloud Provider fails to meet his requirements as described in the contract the user has the choice to terminate the contract. Sometimes, there is term that the Service Level Agreement is subject to the Cloud Providers Policies. The user should ask to read these policies in order to determine whether they are acceptable by him and clarify whether he is bound in cases that these policies change at a later point of time. The Cloud Standards Customer Council has issued a "Practical Guide to Cloud Service Level Agreements" that provides the steps that consumers should take to evaluate cloud Service Level Agreements in order to compare cloud service Providers or negotiate terms with a provider [5].

An additional concern is related to the development of a number of Cloud Computing projects that usually use automated mechanisms for compliance with terms of the Service Level Agreement. The CloudScale project which assists service Providers in analyzing, predicting and resolving scalability issues [6] has guidelines (CloudScale's HowTos) to solve the scalability issues detected and can define the service consumer 's services according to the agreed SLA as well as the Responsibilities of Service Provider to fulfill SLA. In Cloud-TM project a SLA model has been proposed in which when the offer has been accepted by the customer, there is a method to ensure that network equipment's are configured to guarantee the contracted SLA parameters [7]. There are a variety of projects that include as part or final step solutions for the monitoring mostly of the agreed SLAs. Assuming that all requirements of the traditional contract law has been met in order to have a valid contract (e.g. consent, meeting of the minds, acceptance, consideration) there might be an issue of what happens if the parties (Cloud user and Cloud Provider)comply with these methodologies but an error occurs. Who has the responsibility of the system error?

III. WHO CAN SUE THE CLOUDPROVIDER?

The "Privity of contract" principle means that a contract is concluded only between its parties and with no other person as well as the contract can be enforced only by the parties of the contract and by no other person. The "Privity rule" applies to the contract concluded between the Cloud user and the Cloud Provider. Thus, only one of these parties cans sue the other. Although, the doctrine of privity of contract means only that the party of the contract can bring an action on the contract, nevertheless this party is not excluded from the possibility that he may have some other additional causes of action e.g. on tort.

Although, the most common contract is between the Cloud user and the Cloud provider, other contracts may also exist. For example, the Cloud Provider may enter into a contract with a platform Provider in order to use its platform or contract between the Cloud Provider and an infrastructure Provider for the provision of the infrastructure for the services. The contract principles apply to all these situations. And only the parties to these contract can sue each other.

The liability regime of Cloud Providers and consequently who is eligible to claim damages from them are related to the core business of Cloud Providers. The extent of liability of Cloud Provider's is related with only the person or entities that are possibly entitled to claim compensation for damages either due to breach of contract or due to other obligations imposed by other laws. Moreover, as a consequence of the duty of care, the Cloud Provider is liable to whom it owns this duty. Thus, Cloud Providers might also have potential liability under tort law either to its users or any third party who has injured from its behavior like third parties whom their Intellectual property rights have been infringed by the use of the Cloud provider's software. In all cases the determining factor is the wording of the signed contract since it is this fact that is going to determine the magnitude of loss.

IV. JURISDICTIONAL ISSUES

The advert of Cloud Computing created a marketplace that requires a global approach towards a coherent and predictable framework. One of the important elements of Cloud Computing is that data are moving and it is questionable which court can adjudicate a case of potential dispute or which law applies-particularly when data are moving through different jurisdictions. However, there is a need for certainty and predictability for the Cloud Computing contracts since both individuals and businesses want to know the requirements they need to comply with. Compliance is not possible without knowing which laws are applicable to the Cloud contracts as well which law applies to a potential dispute

Therefore, the essential question is how to find the proper Court that has the authority to adjudicate the potential dispute as well as which law regulates Cloud Computing contracts. Jurisdictional principles have been traditionally developed for contracts concluded in the same sovereignty emphasizing to the presence and domicile of the defendants within a particular sovereignty.

Thus, territoriality provides a fundamental determinant for the assertion of personal jurisdiction. Nevertheless, considering the dynamic allocation of Cloud Computing tasks it is questionable whether the traditional jurisdiction principles apply and to what extend can regulate cross border Cloud Computing contracts.

If parties have chosen jurisdiction for the adjudication of their Cloud contract this is valid clause on the contract. Nevertheless, this is questionable when the Cloud Provider contract is presented as standard form contract-click wrap agreement. The forum selection clause might be invalidated on the basis of the inequality of bargain among the parties. In this case, as well as when there is not such clause in negotiated contracts, there are a variety of issues that need to be considered in order to determine the proper applicable law.

Moreover, the issue of personal jurisdiction for Cloud Computing contractual disputes over nonresident defendants is a confusing one. Initially, for the assertion of jurisdiction, the determinant factor was the defendant's present while latter the consent factor was also added. Still these bases for personal jurisdiction has proven to be inefficient, thus the Courts applied the minimum contacts test which examines the relationship between defendant and the forum. This test comes with uncertainties especially when there is only one contract with the forum that needs to satisfy the minimum contacts test. The Courts in order to address the increasingly complicated situations added to the minimum contacts test the reasonableness test. This test requires additionally that the minimum contacts of the defendant with the forum should be of such nature that does not offend traditional notions of fair play and substantial justice

Courts examine additionally, besides the requirements of minimum contacts and reasonableness, whether the defendant has purposely avail himself to the forum, whether his activities are targeted intentionally to the forum as well as whether he has created continuous obligations with the particular forum. If the calculation of these parameters results to the conclusion that the defendant conduct is such that it looks reasonable to anticipate for him to be hauled in the specific forum, then there is basis for personal jurisdiction. In the process of this calculation other factors are considered additionally such as the interest of the forum State to adjudicate the case, the public policies of the forum, and the degree of burden that the decision will create to the defendant.

How these principles apply to Cloud Computing is highly questionable. The unique characteristic of Cloud Computing is the transfer of data through multiple jurisdictions. Each country at the same time has different law applicable to jurisdictional issues. Usually if parties have not decided upon applicable law, then the law of the closest connection with the contract will apply. The proper law in this case should be inferred from the terms, circumstances and all related matters to the contract. This is known as the "inferred test". The thing is how to determine these factors and apply to Cloud Computing contracts. What constitutes minimum contact at Cloud? It looks that there is no uniform rule for the finding of proper law in Cloud Computing contracts. These principles could be applied in analogy to this environment and decided on a case by case basis. In each case the Court should examine the route of the data through the various nodes and the significance of the processing of these nodes in order to determine the applicable node. Further analysis is beyond the scope of this work.

V. CONTRACTING OUT CLOUD USER'S DATA PROTECTION RIGHTS

The Cloud management system provides a platform where users can access data from anywhere in the world [8]. Data and data processing are protected by data protection laws thus exposing the Cloud Provider and the Cloud participants to liability issues. Moreover, the regulation of data protection rights is regulated differently in various jurisdictions. When data are given to the Cloud Provider through the client or an intermediary, the Provider partitions and replicates data in the Cloud infrastructure. Can the client claim that the data is not used for the purpose collected? How can the Provider guarantee that each participating node will use the data only for the specified purposes? Another issue is whether the user (the Cloud client) is entitled to access his/her data held by the Cloud Provider and if it is appropriate to correct or erase such data. Additionally, while the user might have the right to request and obtain access to his/her data from the Cloud Provider, it is not clear whether this access includes revealing information regarding where the data reside and other information regarding the Cloud nodes that store and process the data. Moreover, the data circulation in Cloud Computing is dynamic. The transfer of data is decided based on the availability of nodes that can execute the task at the specific point in time and the replication optimizing the performance of the Cloud at the specific point in time. These are real time decisions that cannot be pre-defined. Also the rules that govern such decisions are part of the logic of the software of the Cloud Provider and many times not publicly available.

In the healthcare sector cloud solutions are becoming more and more appealing, since they can offer significant cost reductions. Health care monitoring systems are systems that are responsible for tracking the health of the patient. The patient himself can insert data collected from heath devices or the devices can be connected to the system and submit data. In each case several data are collected from a variety of patients in different formats. For example collected data could be X- rays, temperature, hear bits, blood pressure etc. When coupled with telemedicine applications [9], the patient monitoring system can provide an integrated remote healthcare service for patient diagnosis and treatment. Cloud computing environment is considered as an approach that allows real-time data accessibility with authentication and real time video streaming to support the teleconference. Moreover, cloud enables the exchange of information between Healthcare providers in an efficient manner.

One of the issues that the Healthcare provider needs to consider when deploying applications in the cloud is that existing solutions do not address the security requirements demanded by the sensitive health care data. Detailed analysis of these issues is beyond the scope of this work. However, a well drafted contract between the Cloud Provider and Cloud user or the contract between the Cloud Provider and the platform or infrastructure Provider could ensure efficient and balanced protection. The contract should facilitate the lawful and fairly circulation of data within the Cloud environment and specify the obligations of the Cloud provider. In a Cloud Computing contract users usually do not object to have the Provider collect and publish cumulative statistics provided that the data cannot be manipulated to obtain information about a specific record or a specific data source.

In cases where data mining algorithms are allowed to execute in the data they should guarantee that the algorithm produces statistics that guarantee privacy.

Furthermore, it seems that a visible solution could be that appropriate software should be provided (by the Cloud Provider to the user) to allow the user to view and update the application data as well as his/her personal data. The software should be able to inform the user regarding the number of nodes that execute his/her application, the number of partitions made of the application data and the number of replicas of the data at the moment of request. Moreover, there is a question whether the Cloud Provider can by contract with the Cloud user waive the users' privacy and data protection rights. If the Cloud user consents to waive his data protection rights does this constitute a valid contract based on the principle of freedom of parties? Does contract law preempt the data protection law? The answer to this question is not straightforward. It has to be addressed in the specific context of the contract and in relation to the specific legislation.

VI. GOVERNMENT OF INDIA AND CLOUD COMPUTING

India government has implemented an initiative called GI Cloud or Meghraj. The focus of this initiative is to evolve a Strategy and implement various components including governance mechanism to ensure proliferation of Cloud in Government[10]. The Government of India has implemented a number of Information and Communications Technologies (ICT) initiatives under the National e-Governance Plan (NeGP), including creation of ICT infrastructure both at the centre and state levels[4,11]. The infrastructure thus created will provide the basis for adoption of cloud computing for the government with the objective of making optimum use of existing infrastructure, thus helping achieve the ultimate goal of NeGP[11].

A task force has been set up to give necessary direction with respect to the various activities which include creation of a detailed plan on the cloud strategy, cloud architecture, cloud implementation plan and roadmap[12]. DeitY set up two committees : the GI-cloud Task Force under the chairmanship of Additional Secretary (e Governance) to propose policy and guidelines for "Government as a user of Cloud Computing" and the Cloud Computing Working Group to work out the recommendations for evolving a comprehensive framework by the Government for adoption of Cloud in the country taking into account the policies and standards relating to jurisdiction, cross-border data flow, data security, data location etc. and other related aspects for enabling cloud services in India[13].

The Jammu & Kashmir state government adopts cloud computing for its e- Governance services. The Government, using the State Data Centers based out of Madhya Pradesh, is provisioning e Governance services such as issuing death or birth certificates and trade licenses through the cloud. The Jammu & Kashmir Government uses Microsoft's solution to implement cloud computing.[14]. The state of Jammu & Kashmir is the first state that utilized cloud computing services[15].

CDAC has established a Private cloud environment to offer basic cloud services such as Infrastructure, Platform, and Software service to Government and SMEs[9]. CDAC has numerous projects at Cloud such as Meghdooth which is free and open source. Cloud stack developed by CDAC Chennai is a one stop solution for implementing in Cloud environment. One of the remarkable activities ongoing is Integration of Private Cloud computing environment with existing Garuda Grid (India's National Grid Computing Initiative)[16].

Moreover, the National Telecom Policy 2012 has recognized that the advent of technologies like cloud computing present a historic opportunity to enhance India's service delivery capabilities to a new level domestically as well globally, enable social networking and m-Commerce at scale which were not possible through traditional technology solutions[17]. The Confederation of Indian Industry, in a report titled "The Indian Cloud Revolution" has stated that there is a need for statutory compliance to laws and regulations. The report concludes that a set of necessary rules and regulation should be created by the Government such as those related to privacy and confidentiality to protect against accidental access to information [18].Cloud computing helps also in good governance since it makes easy the transfer of data and reduce the cost of ICT infrastructure. The adoption of cloud computing enhances the effectives of e-Panchayat as well as it will bring better result in governance especially in rural India[19]. The benefits of adopting cloud services in agriculture sector which contributes 20% to India's GPD and is the biggest employment source it will be a serious attempt to develop rural India. Gujarat Government realized the importance of cloud computing services and has identified the importance of electronic contracts on cloud. There are several contractual constraints to be addressed by the contractual agreements between clients and providers that are not adequately addressed by the cloud computing interface such as the data location and security[20]. An interesting initiative is that of the University of Pune (UoP) that announced it will use cloud computing for its exam systems in four faculties [21]. Part of literature argues that Cloud Computing applications should be employed by Indian Railway because it will allow for the exact calculations of numerous situations that currently cause loss to the Indian Railway [22].

Additionally, the Department of Electronics and Information technology has a FOSS Initiative cell to develop and support Free/ Open Source Software in India[23] with a Free and open source Division[24]. NRCFOSS has come out with BOSS – Bharat Operating system Solutions with support for Indian Languages. Indian Government adopts OS applications e.g. among others, the Government of Tamil Nadu has issued a governmental Order that makes mandatory the installation of BOSS in all computer systems and should be used by Staff Members of Information Technology Department, the Government has launched the IT@ School project in Kerala, the Open government Platform in cooperation with the U.S. Government[25].

Moreover, the Unique Identification Authority of India (UIDAI) [26]has introduced the Aadhaar[27]project which uses open source for software development[28] The principal engineer for Aadhaar, Regunath Balasubramanian, explained [29] that open source became the first choice because technical requirements required vendor neutrality and FOSS helped achieved vendor neutrality which is very important for a initiative for national importance. However Aadhaar is not totally based on open source components and during its implementation to states that completely under open source created problems like interoperability problems. Kerala has found that Aadhaar is problematic and unacceptable since its implementation violates the States FOSS policies[30].

The issue thus, that need to be addressed is how the India Government would be able to combine cloud computing and open source solutions in accordance with its regulatory framework. The open source licenses are not compatible with cloud computing contracts. Thus, a specific legal framework need to be developed to solve these issues.

VII. CONCLUSION

Cloud computing has shifted the internet transactions from a centralized to a distributed environment which enable users to use data and software located in the internet rather their computers or their servers. The employment of Cloud Computing is associated with substantial benefits for its participants such as cost reductions, new business models, new services offered as well as new ways of reaching markets. This phenomenon comes with a set of opportunities as well as major challenges. One of these challenges is its regulation since the legal framework for operating in Cloud computing will be critical in determining the pace of the development of the Cloud.

The analysis of the Cloud computing related contracts affirms that the Cloud Computing establishment and its increased use create concerns due to uncertainties of its legal framework. These uncertainties refer mainly, but not exclusively, to the validity of contracts with the Cloud Provider (including Service Level Agreements) and more specifically with the limitation of liability clauses presented in these contracts. The contracts with the Cloud Providers often include confusing terms regarding who is liable particularly in cases of errors that cause damages. Generally, the enforceability of standard Service Level Agreements are viewed as analogous to click wrap contracts. That view creates a presumption that these contracts are valid as long as the assent of the Cloud user has been affirmed. However, it is unclear whether the Cloud user has given a valid acceptance. The employment of traditional contract principles functions as a patchwork since it cannot address the peculiarities and novel issues raised by the use of Cloud computing. Thus, the current legal framework cannot regulate adequately the Cloud computing environment and therefore a new legislation need to be enacted which will exclusively regulate the legal framework of Cloud computing applications.

A proposed solution might be the enactment of a specific legislation on Cloud Computing by the Indian Government - that legislation will be the first of its kind- establishing a coherent and secure legal framework which will prescribe in a sophisticated and effective manner the Cloud Computing legal environment for both Governmental and Industrial applications

REFERENCES

[1]The Restatement (Second) of Contracts(USA), § 1, Contract Defined
[2]The Indian Contract Act 1872, section 2(h)

[3]W. Kuan Hon, Christopher Millard and Ian Walden., "Negotiating Cloud Contracts: looking at clouds from both sides down", 16 Stan. Tech. L. Rev. (2012), p.81

[4]Lee Badger et al., US Government CloudComputing Technology Roadmap, Volume I, Release 1.o (Draft), High Priority requirements to Further USG Agency CloudComputing Adoption, 2011, http://www.nist.gov/itl/cloud/upload/SP_500_293_volumeI-2.pdf

[5]Cloud Standards Customer Council, "Practical Guide to Cloud Service Level Agreement", version 1.0, http://www.cloudstandardscustomercouncil.org/2012_Practical_Guide_t o_Cloud_SLAs.pdf

[6]www.cloudscale-project.eu

[7]https://www.ict-etics.eu/fileadmin/documents/news/ETICS_white_paper_final.pdf

[8]A. Khan, P. Shaikh, C. Dhembre and S. Gawali, " Cloud Services for Collaborative Web Based Project Management System", 8 International Journal of Computer Science Issues (2011), p.180

[9]P. Matlani, N.D. Londhe, "A cloud computing based telemedicine service," Point-of-Care Healthcare Technologies (PHT), 2013 IEEE , pp.326-330, 16-18 Jan.

[10] Government of India Ministry of Communications and Information Technology, Department of Information Technology, Free and Open Source Software, http://deity.gov.in/content/free-and-open-source-software

[11]Government of India Ministry of Communications and Information Technology, Department of Information Technology, Free and Open Source Software, http://deity.gov.in/content/free-and-open-source-software

[12]Government of India, Ministry of Communications & IT ,Department of electronics and Information Technology "GI Cloud (Meghraj)Adoption and Implementation Roadmap", April 2013

[13]NASSCOM, "Government of India Cloud initiative", http://www.nasscom.in/government-india-cloud-initiative

[14]Arun Chandrasejaran and Mayank Kapoor, Frost & Sullivan 2011-Market Insight, " State of Cloud Computing in the Public Sector- A Strategic analysis of the business case and overview of the initiatives across Asia pacific

[15]The Economic times, "J&K uses MP govt's cloud computing facilities to roloout egovernance", http://articles.economictimes.indiatimes.com/2010-06-25/news/27582997_1_data-centres-cloud-model-cloud-services

[16]Cloud Computing at CDAC, http://www.cdac.in/index.aspx?id=cloud_ci_cloud_computing

[17]Government of India, Ministry of Communications & IT ,Department of electronics and Information Technology, "Government of India's GI Cloud (Meghraj)Strategic Direction Paper", April 2013, http://deity.gov.in/sites/upload_files/dit/files/GI-Cloud%20Strategic%20Direction%20Report(1).pdf

[18]Confederation of Indian Industry, " The Indian Cloud Revolution", http://www.cii.in/cloudreport

[19]Chasura R.S. et al., "Cloud Computing : future Buzz for Rural India", Wayamba Journal of Animal Science, (2012), p. 255

[20] An e-governance bulleting from Gujarat Informatics LTD., "Cloud Computing", 7 GIL, 2010

[21] Ardhra Nair, " UoP to introduce cloud computing in four faculties", The Indian Express, 2013, http://www.indianexpress.com/news/uop-to-introduce-cloud-computing-in-four-faculties/1100723

[22]Gaurav Bhatia et.al., " Implementation of cloud Computing Technology in Indian Railway", 37 IPCSIT (2012), p.84, International Conference on Information and Network Technology (ICINT 2012)

[23] Government of India Ministry of Communications and Information Technology, Department of Information Technology, Free and Open Source Software, http://deity.gov.in/content/free-and-open-source-software

[24]Government of India Ministry of Communications and Information Technology, Department of Information Technology, Free and Open Source Software, http://deity.gov.in/content/free-and-open-source-software

[25] Open Government Platform, http://www.opengovplatform.org/

[26]Identification Authority of India, Planning Commission, Government of India, http://uidai.gov.in/

[27] Unique Identification Authority of India, Planning Commission, Government of India, Aadhaar, http://uidai.gov.in/aadhaar.html

[28] PK Jayadevan, " UID: Due to the technology challenges of speed and scale, Aadhaar is an objectof attention", The Economic Times, 2012, http://articles.economictimes.indiatimes.com/2012-02-07/news/31034068_1_aadhaar-project-unique-identification-authority-biometric-database

[29] Vandana Sharma, "Aadhaar: A Testimony to Success of FOSS in India", Linux for you 2011, http://www.linuxforu.com/2011/12/aadhaar-testimony-to-foss-success-in-india/]that

[30] Deepa Kupur, "Aadhaar software locked in with "Windows", The Hindu, 2010, http://www.thehindu.com/news/national/aadhaar-software-locked-in-with-windows/article863657.ece

SLA-aware Provisioning and Scheduling of Cloud Resources for Big Data Analytics

Mohammed Alrokayan, Amir Vahid Dastjerdi, and Rajkumar Buyya
Cloud Computing and Distributed Systems (CLOUDS) Laboratory,
Department of Computing and Information Systems,
The University of Melbourne, Parkville, Victoria 3010, Australia
Emails: m.alrokayan@student.unimelb.edu.au, amir.vahid@unimelb.edu.au, rbuyya@unimelb.edu.au

***Abstract*—The stunning growth in data has immensely impacted organizations. Their infrastructure and traditional data management system could not keep up to scale of Big Data. They have to either invest heavily on their infrastructure or move their Big Data analytics to Cloud where they can benefit from both on-demand scalability and contemporary data management techniques. However, to make Cloud hosted Big Data analytics available to wider range of enterprises, we have to carefully capture their preferences in terms of budget and service level objectives. Therefore, this study aims at proposing a SLA and cost-aware resource provisioning and task scheduling approach tailored for Big Data applications in the Cloud. Current approaches assume that data is pre-stored in cluster nodes prior to deployment of Big Data applications. In addition, their focus is purely on task scheduling, and not virtual machine provisioning. We argue that in the Cloud computing context this is not applicable, because the nodes are provisioned dynamically (data cannot be pre-stored) and leaving provisioning to user may lead to under or over provisioning that can both lead to SLA or budget constraint violations. Therefore, in this study we first model the user request, which consist of Big Data analytics jobs with budget and deadline. Then, we model infrastructures as a list of data centers, virtual machines (offered in a pay-as-you-go model), data sources, and network throughputs. After that, to address the aforementioned issues, we propose and compare cost-aware and SLA-based algorithms which provision cloud resources and schedule analytics tasks.**

Keywords*—*Cloud Computing, Big Data Computing, Big Data.

I. Introduction

With prodigious growth in Web and social network data, more than ever before, it is clear that big data is coming. The big data wave is going to surface new opportunities for enterprises although constitutes series of challenges. The key issues is how to ingest Big Data and convert it to information and knowledge that possesses business value. However, this conversion is not economically viable for small to medium enterprises in a traditional infrastructure setting.

Cloud computing is growing rapidly as an extremely successful paradigm offering on-demand infrastructure, platform and software services to end users. Cloud computing, with its on-demand elasticity, and pay-as-you-go model, enables data intensive applications to dynamically provision resources and process large data sets in parallel, which was not economically feasible in a traditional data management systems. To pave the way for organizations to adopt Cloud-hosted Big Data analytics, we have to carefully consider their preferences in terms of budget and service level objectives through provisioning and scheduling phases, which is the focus of this study. There are three main deployment models in cloud computing: Private Cloud, Public Cloud, and Hybrid Cloud [1]. Our focus is on Hybrid Clouds, where applications are deployed and run across private and public Clouds in seamless manner.

There is no single tool that provides a complete solution for Big Data analytics. We use the Lambda architecture [2] to tackle the problem of Big Data computing via three layers, namely batch, serving, and speed layer. In this paper, we restrict our focus to the batch layer that is responsible for running a function on a the whole dataset to build batch views which are indexed and later are utilized by other layers to compute the final query result. Scalability, simplicity, and fault-tolerance are among desired properties of the batch layer. We utilized MapReduce [3] for the batch layer as it possesses the aforementioned properties and is capable of processing large set of data in parallel to overcome disk I/O bottlenecks.

The majority of SMEs are constantly looking for cutting edge technologies and solutions to accomplish objectives of the company more efficiently and at the minimum cost and big data processing via Cloud resources is not an exception. There are many SLA-based Big Data computing and MapReduce scheduling studies [4], [5], [6], [7] in the Cloud context, however they do not provision Cloud resources dynamically. Instead, they have the resources already pre-provisioned (static) on a private Cloud, which form a virtual cluster. We argue that Cloud resources should be provisioned dynamically and on-demand based on the application workload and the size of the data. This introduces new challenges, namely: a) how many and which type of cloud resources to provision; b) which private infrastructure or public cloud provider to select for a given request with budget and deadline constraints; and c) given that data is geographically distributed, which resources should be chosen that minimize the data transfer and processing costs.

Our major contributions are summarized as follows: 1) a model for SLA-based resource provisioning and tasks scheduling for Big Data processing in cloud environments; 2) an SLA-based and cost minimization algorithm to provision cloud resources and schedule MapReduce tasks in the batch layer of Lambda architecture, and a technique for reducing the size of the search space; 3) an approach to enable the algorithm to run in parallel taking the advantage of multi-core system to find an optimal solution; and 4) the design and development of a new extension to CloudSim to evaluate and compare the algorithms.

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

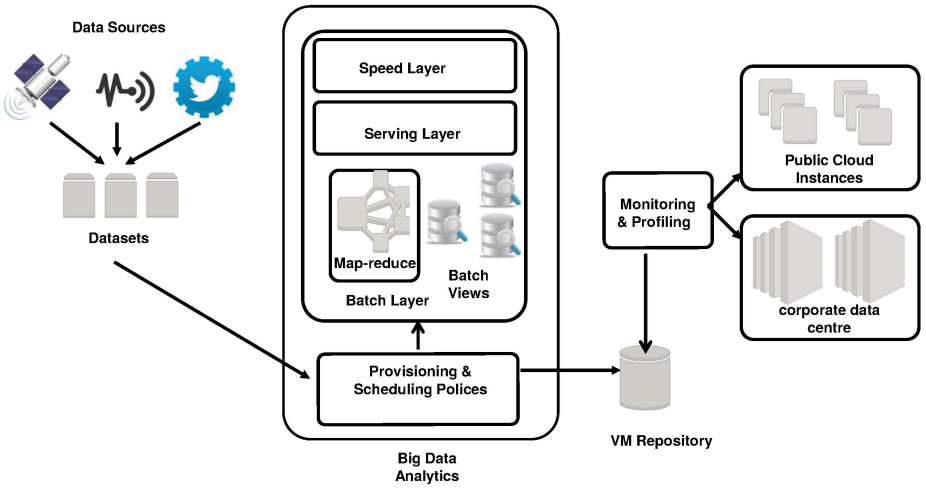


Fig. 1. The Proposed Architecture for Cloud-hosted Big-Data Analytics

The rest of the paper is organized as follows: In the next section, we position our work in the context of Cloud-hosted Big Data analytics while we are describing the proposed architecture. Section III presents the MapReduce on cloud model for the batch layer and discuses formulation of the problem, objectives, and constraints. Next, the proposed algorithm is described in Section IV following by performance evaluation of the algorithms and experiments' results in Section V. We present related works and compare them with our approach in section VI. We conclude the paper in Section VII with ideas on future directions.

II. Architecture

The proposed architecture is depicted in Figure 1. It is based on the Lambda architecture [2] and its main components are explained below:

Data Sources and Datasets- There are more objects than humans connected to the Internet, and their numbers are growing rapidly. These objects are called **Data Sources** and can send what they have sensed from different locations and environments. Datasets are raw data collected (eg. sensors and social media feeds) from data sources and are source of truth.

Batch Layer- When it comes to Big Data analytics, executing a random query on the whole dataset in real time can be computationally expensive. This problem can be alleviated by the use of batch views and pre-computed results to speed up query execution. The role of Batch Layer is to generate batch views from datasets.

Map-Reduce Component- MapReduce On Cloud is an attractive model for enterprises to build batch views due to its flexibility, agility, and low cost. Apache Hadoop is among the most popular implementations of MapRedcue. However, the proposed model in this study is not Hadoop compatible. The reason is Hadoop's schedulers are designed for static cluster of homogeneous machines in a single datacentre, while our model considers heterogeneous virtual machines (VM) across clouds. MapReduce consists of three phases: Map, Shuffle and Sort, and Reduce. MapReduce data is presented in key-value pairs for the mappers in the map phase for processing. Each mapper process a block of key-value pairs, and the size of the block is defined by the user. Mappers emit processed data in key-value pairs, generating intermediate data for the reducers in the reduce phase to aggregate the values. Shuffle and sort phase groups the values with the same key, and sort the keys. Reducers receive intermediate data with a set of values for each key for aggregation, and send the result as batch views. We consider the following characteristics for MapReduce in our architecture : 1) The reduce phase can not start before all map tasks finished. 2) Map tasks are almost homogeneous because they have the same function and almost same data block size. 3) Reduce tasks are heterogeneous because each reduce task process different size of data based on the emitted data (intermediate data) from the map phase. 4) A reduce task can be scheduled in the same map node to reduce the intermediate data transfer time and cost between nodes. 5) Completed map tasks can start sending intermediate data to reduce nodes even before all map tasks finished. This can minimize network I/O bottleneck and to save on execution time as well.

Serving Layer and Speed Layer- The Serving layer typically consists of a database system which consumes batch views and facilitates complex queries. To this extent, we are capable of executing queries on the precomputed views, however to execute an arbitrary query on an evolving dataset in real time, we need the Speed Layer. Resource Scheduling and Provisioning policies for these two layers will be presented in our future works.

Monitoring and Profiling- This component consists of a collection of monitoring services that, together with benchmarking toolkits, extract and analyze performance statistics about public and private Cloud instances for a running or a benchmarking application. Profiling occurs while Data procession is running, or separately for a new version of application or instance to update and improve the collected statistics.

Resource Scheduling and Provisioning Component- In order to enable cloud-hosted MapReduce for application with SLA, we require the Cloud resource provisioning and scheduling component, which is a focus of this study. This component make decisions on both how many and which type of VM are required (provisioning phase) and which task

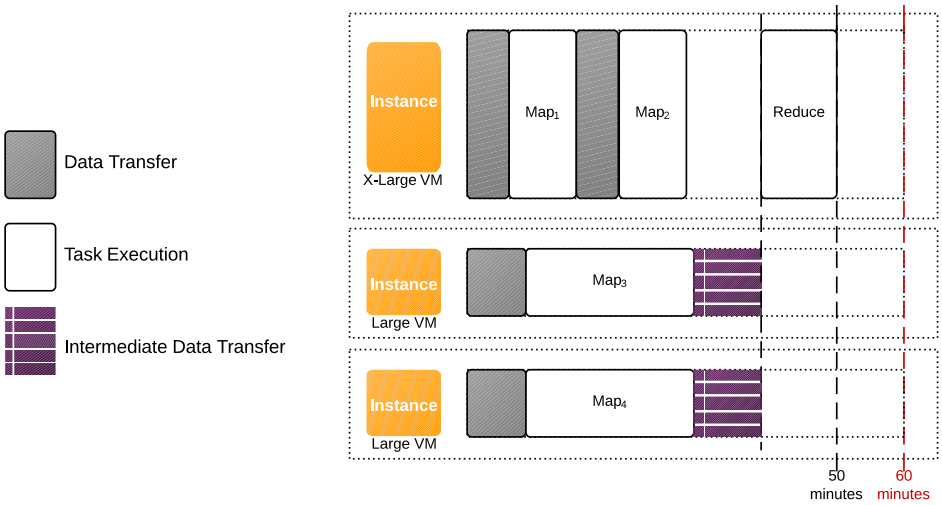


Fig. 2. An example of scheduling four map tasks and one reduce tasks in three virtual machines to achieve the deadline of 60 minutes

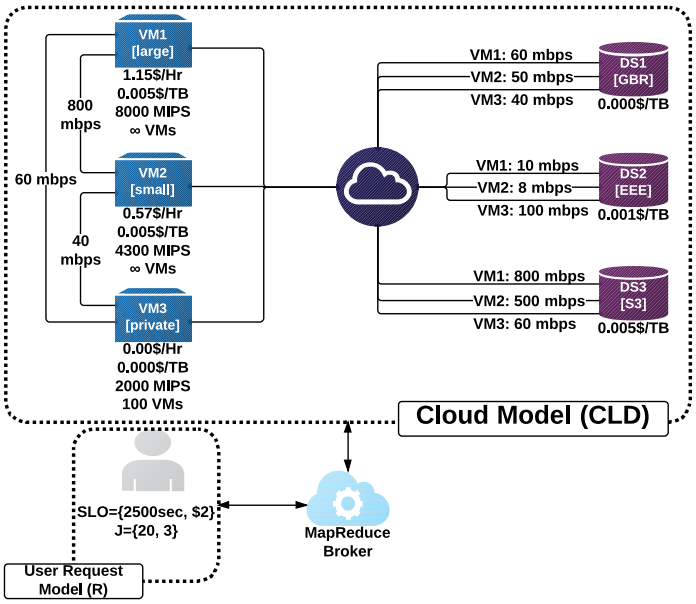


Fig. 3. Cloud (CLD) and User Request (R) Models

is running on which resource (scheduling phase) to meet the deadline for creating batch views as well as minimizing the cost. An efficient scheduling and provisioning component can guarantee higher level of accuracy for Big Data analytics by ensuring that there are always batch views built in time for Serving Layer. Figure 2 demonstrates an example of provisioning three resources (virtual machines) and running a MapReduce job with four map tasks and one reduce task on a Cloud within 50 minutes, where the deadline is 60 minutes.

III. System Model

There are two models in our system: Cloud Provider (CP) and User Request (R) Models. The Cloud model (shown in Equation 1) consist of: a set of data centres (DC), a set of data sources (DS), a matrix (T_{VM}) that shows throughputs between a virtual machine type and other types of all data centres in megabits per second (mbps), and a matrix (T_{DS}) that shows throughputs between a data source and a virtual machine type in megabits per second (mbps). Each DS has a cost for transferring the data from it (CT_{DS}) per terabyte. Each data centre has a set of virtual machine types (VM). Each virtual machine type has: cost of leasing (C_{VM}) per hour, the cost of transferring the data from it (CT_{VM}) per terabyte, the performance of the virtual machine ($MIPS$) in million instructions per second.

$$CP = \{[DS], [DC], [[T_{VM}]], [[T_{DS}]]\} \quad (1)$$
$$DC = \{[VM]\} \quad (2)$$
$$VM = \{C_{VM}, CT_{VM}, MIPS\} \quad (3)$$
$$DS = \{CT_{DS}\} \quad (4)$$

User request (R) model (shown in Equation 5) consists of: SLA objectives (SLO) and MapReduce Job (J). Each SLO includes: Budget (B) and Deadline (D).

$$R = \{SLO, J\} \quad (5)$$
$$SLO = \{B, D\} \quad (6)$$

A MapReduce Job (J) (shown in Equation 7) consists of a DS, a set of map tasks (M_{Task}), and reduce tasks (R_{Task}). Each M_{Task} consists of: Input Data Size ($DSize$), the required million CPU instructions(MI), and the size of the intermediate data ($IDSize_{R_{Task}}$) to each reducer R_{Task}. Each R_{Task} has the required instructions in million instructions (MI). The optimization algorithm receives a J as a part of R.

$$J = \{DS, [M_{Task}], [R_{Task}]\} \quad (7)$$
$$M_{Task} = \{DSize, MI, [IDSize_{R_{Task}}]\} \quad (8)$$
$$R_{Task} = \{MI\} \quad (9)$$

The objective is to satisfy the SLA requirements of the user while minimizing the total cost. The total cost of running the MapReduce job is denoted as TC. Given that total of n machines and m data sources are used, the TC can be computed as shown in Equation 10 where LP is the leasing period for a virtual machine and TD_{VM} and TD_{DS} are total data in terabyte transferred from a machine and a data

source respectively. The total execution time for running the MapReduce job is denoted as: ET, which is the total time of executing a task and transferring the data in and out from data centres.

$$TC = \sum_{i=1}^{n} C_{VM_i} * LP_{VM_i} + CT_{VM_i} * TD_{VM_i} + \sum_{j=1}^{m} CT_{DS_j} * TD_{DS_j} \quad (10)$$

As descried in Section III, the SLA requirements consist of Budget (B) and Deadline (D). As a result, algorithms' objective is given as:

$$Min\,(TC)\; Subject \quad to \quad TC < B \quad and \quad ET < D \quad (11)$$

IV. Algorithms

The provisioning and scheduling problem described in the last section is a multidimensional knapsack problem that was shown to be NP-complete. To tackle the problem, one may consider a greedy algorithm [8]. However, it cannot be directly adopted as it is not capable of satisfying the budget constraint. In addition, it is important to emphasize that the optimization algorithms are required to both determine what is the best set of Cloud resources to provision and also how to schedule tasks on those resources. Knowing the the characteristics of the problem, series of algorithms are presented and later in Section V their performances are compared. Algorithms are: List and First Fit sorted by Cost (LFFCost), Backtracking sorted by Cost (BTCost), Branch and Bound sorted by Cost (BBCost), Branch and Bound sorted by Cost and Performance (BBCostPerf), Branch and Bound with Multiple trees sorted by Cost (BBMultiCost), and finally Branch and Bound sorted by cost based on a Pruned Tree (BBPruned), which will be describe in detail.

The LFFCost algorithm [5] is a heuristic that is expected to have lower execution time. However, it can not handle budget and deadline constraints. The rest of the algorithms (BTCost, BBCost, BBCostPerf, BBMultiCost, and BBPruned) are tree-based which can cover all possible solutions. Prior to describing the algorithms we would like to describe two types of trees that are used by the aforementioned algorithms: Standard Tree and Pruned Tree.

A. Standard Trees

Figure 4 illustrates an example of a constructed Standard Tree for two tasks (Task#1 and Task#2) and two VM types (**L** for Large VM and **XL** for X.Large VM). The depth (levels) of the tree is the total number of map and reduce tasks (Task#1 and Task#2 in Figure 4), while the breadth (branches) is the total number of tasks multiplied by the number of VM instances. Therefore, in each level of the tree, each node is a VM type that can be selected for a task execution. The number of branches of all nodes are the same, which is the number of tasks (map and reduce tasks) multiplied by the number of VM instances. The reason why the Standard Tree is constructed in this way is that we need to cover all of the possibilities for scheduling MapReduce tasks. For example, in Figure 4 we have two tasks and two type of VMs. As a result, we have four branches under each node. This can be considered as a disadvantage in the Standard Tree as it grows exponentially as number of tasks and VM instances increases.

Standard Tree is sorted by cost ascending from left to right, so it will start consolidating all MapReduce tasks into the cheapest virtual machine, which is the most left leaf solution. If that solution does not satisfy the deadline constraint; it schedule one of the tasks to the next cheapest virtual machine. However, if the budget is violated the traversing process will stop with no solution found. An example of a solution set/vector is the the third leaf node in Figure 4 (v = $\{L_1, XL_1\}$), which means that the first task will be scheduled in a Large VM L_1, and the second task in an X.Large VM XL_1.

B. Pruned Tree

Figure 5 illustrates an example of a constructed Pruned tree for two tasks (Task#1 and Task#2) and two types of VM (**L** for Large VM and **XL** for X.Large VM). We managed to reduce the size of the tree compared to the Standard Tree. Similar to the Standard Tree, the depth (levels) of the tree is the total number of map and reduce tasks. However, the breadth in Pruned Tree is different than the Standard Tree. As shown in Figure 4 and 5, L2 is eliminated from children of the root node. The reason is that there is no difference in cost and execution time of scheduling the first task in L1 compared to L2. Similarly, XL2 (and generally for root children, all instances of each specific VM Type except one) is eliminated. The rest of the nodes are built from: 1) the set of nodes in the path from the root to the current node, and 2) an extra VM instance from each VM type, where the maximum number of VM instances to be added from each type is the number of MapReduce tasks. As illustrated in Figure 5, node branches are not of the same root branches, not like Standard Tree branches. The $path$ will be selected as an optimal solution once it does not violate the SLA constraints. In summary the solution space of Pruned Tree is considerably smaller in size compared to Standard tree, as out of VM instances with similar performance, we have only kept one and removed the others when it makes no difference in total cost and execution time.

Traversing the Pruned Tree is similar to the Slandered Tree, as nodes are sorted by cost ascending from left to right. Hence, it will start consolidating all MapReduce tasks into the cheapest VM (i.e the most left leaf solution). If that solution does not satisfy the deadline SLA objective; it moves one task to the next cheapest virtual machine, and so on. However, if the budget SLA objective is violated at any time the traversing will stop and return no solution is found. An example of a solution set/vector is the fifth leaf node in Figure 5 (v = $\{XL_1, XL_1\}$) - the first task and the second task will be scheduled in the same X.Large VM XL_1.

C. Algorithms for Provisioning and Scheduling

We propose several algorithms including a modified version of branch and bound algorithm and backtracking for the problem. As described in Section III, algorithms aim at satisfying SLA constraints (budget and deadline) while

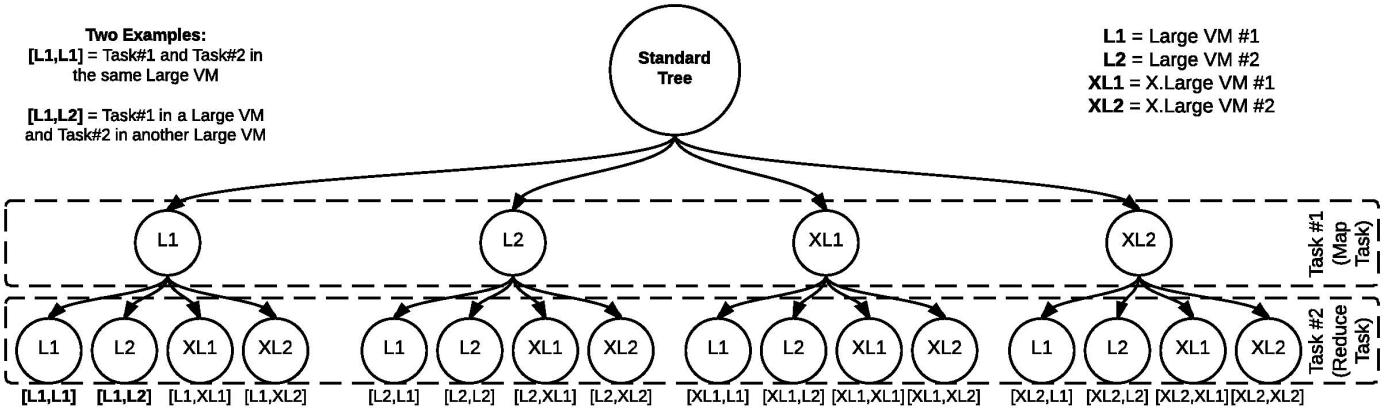


Fig. 4. An Example of a constructed Standard Tree with two tasks (Task#1 and Task#2) and two types of VMs (**L** for Large VM and **XL** for X.Large VM). BTCost, BBCost, BBCostPerf, and BBMultiCost algorithms use this type of tree

minimizing the cost. When algorithms employ the standard tree,it takes long time (days) to traverse the tree and find an optimal solution, especially for certain deadlines and low budget. Therefore, we have limited the maximum running time for algorithms to three minutes and chosen the best solution (even if it is not the optimal one). For all algorithms, solutions are stored in vectors (v). The vector index is a task and the vector value is a virtual machine.

In the branch and bound (BB) algorithms (BBCost, BBCostPerf, BBMultiCost, and BBPruned), we use two functions to estimate the execution time and cost: 1) $GetT(v)$ function: returns the execution time for a given solution vector. 2) $GetC(v)$ function: returns the cost for a given solution vector.

- **LFFCost** - an implementation of the List and First Fit (LFF) algorithm proposed by Hwang and Kim [5]. Virtual machines (VMs) in LFFCost are sorted by cost. It requires two steps for any algorithm to run MapReduce jobs on Cloud: resource provisioning and tasks scheduling. However, Hwang and Kim skipped the resource provisioning step and jumped into tasks scheduling. Therefore, we feed LFFCost with all possible combinations of VMs to compare it with our proposed algorithm.
- **BTCost** - a backtracking algorithm traversing the whole Standard Tree. It checks the execution time and cost on all leafs, and returns the solution with minimum cost that does not violate the deadline.

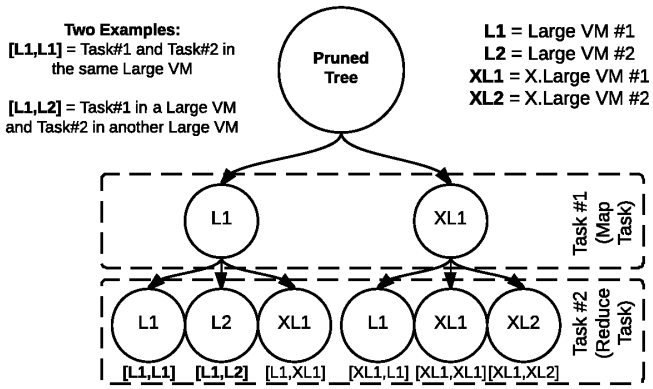


Fig. 5. An example of pruned tree used in BBPruned algorithm with two tasks (Task#1 and Task#2) and two types of VMs (**L** for Large VM and **XL** for X.Large VM)

Algorithm 1: Branch and bound algorithm on Standard Tree sorted by cost

input : Request $(R) = \{SLO, J\}$
output: Scheduling vector v

```
1  v ← 1 ;
2  done ← false ;
3  while !done do
4  |  T ← GetT(v) ;
5  |  C ← GetC(v) ;
6  |  if T < D & C < B then
7  |  |  if is in leaf then
8  |  |  |  return v ;
9  |  |  else
10 |  |  |  v ← GoDeep(v) ;
11 |  |  end
12 |  else
13 |  |  v ← GoNextOrBack(v) ;
14 |  |  if v has no values then
15 |  |  |  done ← true ;
16 |  |  end
17 |  end
18 end
19 return no solution is found ;
20 GoDeep (v)
21 |  return v + 1 ;
22 end
23 GoNextOrBack (v)
24 |  if last element of the branch then
25 |  |  return v without the last value of the vector ;
26 |  else
27 |  |  return v with last value of the vector increased
   |  |  by one ;
28 |  end
29 end
```

Algorithm 2: Branch and bound algorithm on a Pruned Tree sorted by cost

```
   input : Request (R) = {SLO, J}
   output: Scheduling vector v
1  foreach vm from each type of VM do
2  |  vectors ← vectors+ Search({vm},null) ;
3  end
4  return Min(vectors); Search (currentBranch, path)
5  |  foreach vm of currentBranch do
6  |  |  newPath ← path + vm ;
7  |  |  T ← GetT(newPath) ;
8  |  |  C ← GetC(newPath) ;
9  |  |  if T < D & C < B then
10 |  |  |  if is in leaf then
11 |  |  |  |  return newPath ;
12 |  |  |  else
13 |  |  |  |  nextBranch ← newPath ;
14 |  |  |  |  Add to nextBranch one VM instance
   |  |  |  |  from each type ;
15 |  |  |  |  return Search(nextBranch, newPath)
   |  |  |  |  ;
16 |  |  |  end
17 |  |  end
18 |  end
19 end
```

BTCost is a Depth First Search (DFS) algorithm on the described Standard Tree.

- **BBCost** - uses branch and bound to improve the backtracking algorithm (BTCost) by using a validation function on each node (bounding step) to decide whether to go deep on the tree (branching step) or discard the rest of the tree (pruning step). This validation function calculates the execution time ($GetT(v)$) and cost ($GetC(v)$) and compares it with the defined SLA constraints. Algorithm 1 shows the pseudo code of the branch and bound algorithm that is used in BBCost, BBCostPerf, and BBMultiCost. BBCost uses the Standard Tree sorted by cost and initialize the vector v with the value of $v = \{1\}$, which is the cheapest VM. Then the algorithm keeps going deep in the tree unless the solution vector v violates the constraints. After the completion of every major branch (major branches are branches under the root), one of following occurs: 1) If there is a budget violation in any point during the search, the algorithm returns: a) "low budget" if no solution is founded, or b) the solution just before the violation. 2) If the budget is not violated, it goes to the next major branch until a solution which can meet the deadline is found.

- **BBCostPerf** - runs two trees at the same time, one sorted by cost (CostTree) and the other one by performance (PerfTree). CostTree uses BBCost algorithm to find optimal solution, while PerfTree uses an algorithm similar to BBCost but the tree is sorted by the performance of VMs. Our model in Section III shows the measurement of the VM performance is based on Million Instruction Per Second (MIPS). As soon as PerfTree finds a candidate solution, it will be sent to CostTree. CostTree will check its best solution up to this point with the one that has been received by PerfTree. If PerfTree's solution costs less and satisfies both of the budget and deadline objectives, then PerfTree's solution will be taken as the best solution. On the other hand, having another tree sorted by performance (PerfTree) helps us to find out whether there is a feasible solution at all or not (if deadline is violated before finding a solution) earlier than BTCost and BBCost.

- **BBMultiCost** - is an improved version of BBCostPerf. It splits the CostTree into multiple equal parts to search them in parallel along with PerfTree. The number of CostTree trees that runs in parallel depends on the capacity of the machine that runs BBMultiCost algorithm. In general, we based it on the number of cores considering one core for the main application/thread and another core for PerfTree tree. In our case, we leveraged the Hyper-Threaded technology on our machine, which allows us to run two trees in parallel on each core.

- **BBPruned** - described in Algorithm 2 and stands for **B**ranch and **B**ound algorithm for a **Pruned** Tree. It uses branch and bound algorithm on a Pruned Tree to find an optimal solution. BBPruned uses a recursive function ($Search$) to find a solution. The $Search$ function is called on each node except leaf nodes and its inputs are the current branch nodes ($currentBranch$) and the $path$ solution vector. The core functionality of this recursive function is to iterate on each node of $currentBranch$ and return $path$ as an optimal solution if it is a leaf node and it does not violate the constraints, or call $Search$ if it is not a leaf node and it does not violate the constraints. However, the node will simply be skipped if it violates any of constraints. To find a solution with the minimum cost, the Pruned Tree is sorted by cost ascending from left to right, so it will start consolidating all MapReduce tasks into the cheapest virtual machine, which is the most left leaf solution. If that solution does not satisfy the objectives, it evaluates the next cheapest solution.

V. Performance Evaluation

We have extended CloudSim [9] to build a model for a Cloud-hosted Big data analytics environment. For the batch layer, it is worth mentioning that the model is not compatible with Hadoop because Hadoop's scheduler is not designed for Cloud computing and it can not schedule tasks on heterogeneous VMs. VM types that are used in simulation[1] are similar to Amazon AWS VM types and OpenStack private Cloud VM flavours. We performed three sets of experiments as follows:

- **SLA Violation:** Table I listed the SLA violation percentage for the algorithms. The table shows that BTCost has 50% SLA violation for the scheduled

[1]The data, workloads, experiment results and algorithms implementations are available at: https://github.com/Cloudslab/CloudSimEx/tree/master/cloudsimex-mapreduce

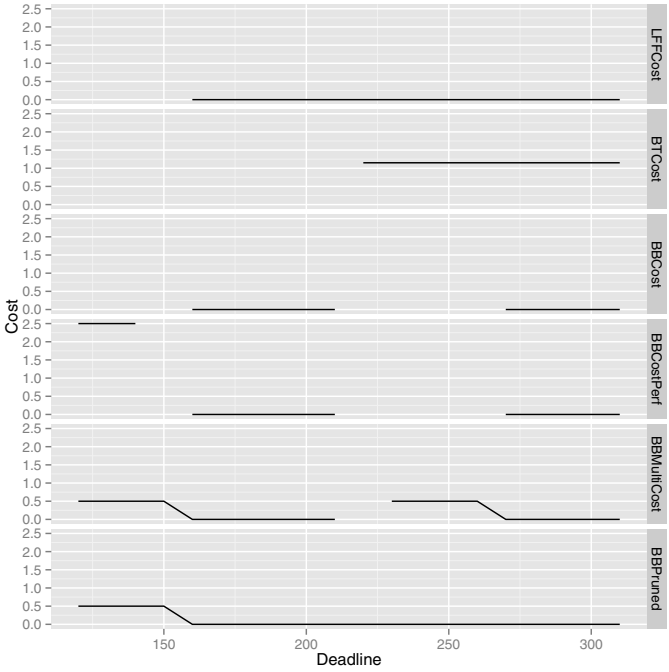


Fig. 6. The cost in dollar for provisioning and scheduling MapReduce jobs on cloud using different algorithms on different deadlines (solid line = SLA satisfied)

TABLE I. SLA VIOLATION PERCENTAGE FOR DIFFERENT ALGORITHMS

Algorithm	SLA Violation Percentage
LFFCost	20%
BTCost	50%
BBCost	45%
BBCostPerf	30%
BBMultiCost	5%
BBPruned	0%

MapReduce jobs, which is the highest reported percentage. BBCost and BBCostPerf come next with a slight decrease in SLA violation percentage. The next algorithm is LFFCost, which violates 20% of the SLA constraints. BBMultiCost shows a good performance with about 5% SLA violation. Finally, this experiment shows BBPruned algorithm causes no SLA violation, and manages to meet MapReduce jobs deadline within the defined budget. That is because BBPruned is capable of traversing the tree faster than the others, and therefore can find a feasible solution within the time limit (three minutes).

- **Cost Minimization:** Figure 6 shows the performance of algorithms and their total cost of provisioning and scheduling with different deadlines. In this experiment the budget is fixed to $2.5 and the deadline varies from 120 seconds to 310 seconds. When an algorithm fails to find a solution within the defined budget and/or deadline, no value for the cost has been reported. We have noticed that BBCost, BBCostPerf, and BBMultiCost manage to find solutions on loose deadlines or tight deadlines. This is because when a deadline is tight, the bounding step prunes most of the tree due to SLA violation, and as the solution space becomes smaller. As a result the algorithms manage to find a solution within the defined period of time. When we have loose deadlines the algorithms manages to find a feasible solution quicker due to the higher probability of spotting a solution at the early stage of search. Figure 6 shows that BBPruned outperforms the other algorithms and minimizes the cost without violating the SLA.

- **Algorithm Running Time:** Figure 7 shows the algorithm running time when deadline varies from 120 seconds to 310 seconds. BBPruned algorithm manages to find solutions in comparability short period of time similar to LFFCost. BTCost (the backtracking algorithm) takes considerably longer time to find a solution on tight deadlines. However, BBCost, BBCostPerf, and BBMultiCost (branch and bound algorithms) do not following a consistent trend. Figure 7 shows the variations for different deadlines- from less than one second to three minutes (which is the maximum allowed running time). We notice that when the deadline gets looser majority of the algorithms managed to find solution relatively faster. The reason behind this variations of running time for the cases of BBCost, BBCostPerf, and BBMultiCost (branch and bound algorithms) is that for some deadline, especially for tight deadlines, the bounding step prunes most of the tree and the algorithms manage to find a solution within the defined period of time. In addition, when we have loose deadlines there is a higher probability of spotting a solution at the early stage of the search. As shown in Figure 7, BBCostPerf algorithm runs faster than BTCost and BBCost because it stops traversing the tree in early stages when PerfTree finds a solution with less cost earlier than CostTree.

VI. RELATED WORK

Most existing MapReduce schedulers and frameworks do not consider budget as a constraint [4], [10], [6], [5], [7], [11], although they help in efficiently running a MapReduce job in Cloud environments. In addition, the majority of the implementations are Hadoop-based [2] and use Hadoop default scheduling algorithms to schedule jobs on Clouds [4], [10], [6], [11]. Nevertheless, Hadoop scheduling algorithms were designed for clusters of homogeneous machines, which is not applicable for Cloud heterogeneous resources. However, the proposed model in this paper is designed for provisioning and scheduling MapReduce applications across instances of multiple clouds.

In addition, majority of works [4], [10], [6], [5], [7], [11] do not consider the time it takes to upload data to Cloud when modelling the scheduling and provisioning problems. Conversely, we consider the time it takes to transfer the input data to the mappers and intermediate data to the reducers. The transfer time is computed based on the size of data and network throughput.

[2]Apache Hadoop: http://hadoop.apache.org/

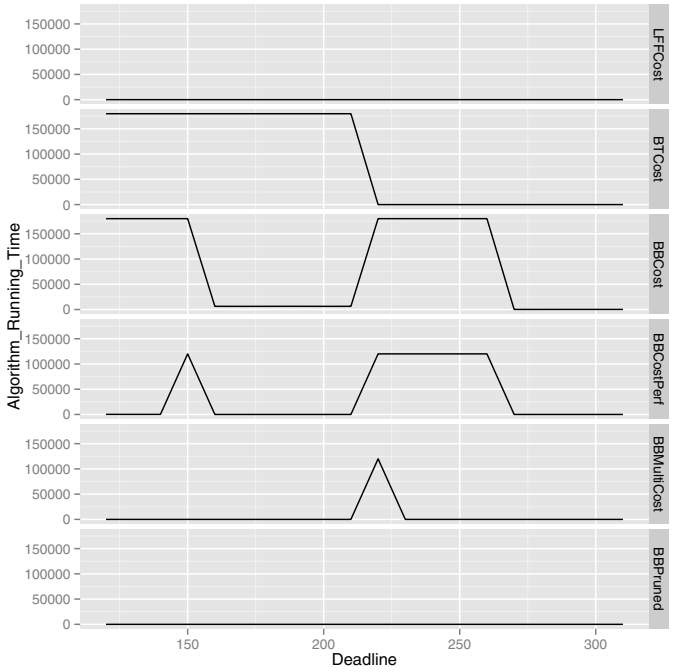


Fig. 7. The running time for the six algorithms in seconds

Lama et al. [4] developed a Hadoop auto-reconfiguration and Cloud resource provisioning system called AROMA. It has two main operations modes: offline and online. The offline mode uses a machine learning techniques to cluster Hadoop jobs to feed the online operations. The online mode operations uses optimization to allocate resources and configure Hadoop. They used Hadoop's default scheduler to run MapReduce jobs on Cloud, which is designed to schedule tasks on Cluster. Hwang et al. [5] proposed a deadline-constrained MapReduce scheduler on Clouds. Their scheduling algorithms skip the resource selection and assume that the user will provide those resources (VMs) to the scheduler. Lee et al. [6] built a system called TARA (Topology-Aware Resource Allocation). TARA does not consider any constraints on budget and deadline. It aims at minimizing the execution time of running MapReduce jobs on Clouds. TARA does not play any role in allocating and provisioning resources on Clouds. Instead, it manages the virtual machine placement. In other words, TARAs goal is limited to VM placement by mapping them to hosts in a datacentre.

We have identified three different MapReduce Cloud infrastructure types in the literature: 1) *MapReduce Cluster on Cloud:* This infrastructure uses virtualization technology to pre-provision a static set of homogeneous virtual machines (VMs), and run MapReduce jobs on all of them. The major contribution in this type of infrastructure is limited to task scheduling, ignoring resource selection and provisioning [6], [7], [11]. 2) *MapReduce Grid on Cloud:* Likewise, in this category virtualization technology is used to pre-provision resources. However, in this category they are shared between users, and they are not destroyed and re-provisioned for each user. The focus here is on fair resource distribution [4], [5]. 3) *MapReduce on Cloud:* The use of a Cloud Infrastructure as a Service (IaaS) to dynamically provision resources is the focus of this category. The contribution here is designing efficient resource provisioning and task scheduling [7].

VII. Conclusions and Future Work

In this paper we discussed the problem of provisioning and scheduling heterogeneous Cloud resources for Big Data analytics and narrowed our focused to Batch layer. We presented an efficient architecture and algorithm (BBPruned) to provision resources and schedule MapReduce tasks in Batch layer for users with SLA constraints (budget and deadline). The proposed algorithm is a modified version of branch and bound algorithm that traverses a customized Pruned Tree, which is smaller than a Standard Tree. The designed algorithm splits the tree into almost equal branches and traverses them in parallel to utilize multi-core systems. Finally, we have shown the efficiency of the proposed algorithm and its ability to generate more prominent solutions faster without violating the SLA constraints. This allows to build batch views in timely manner for Serving layer and bring us one step closer to build a SLA-aware Big Data analytics solution in Cloud.

For future work, we will develop algorithms for the other layers and for private clouds with different classes of users to decrease SLA violation for users with higher priorities. In addition, we are going to investigate a hybrid Cloud scheduling and provisioning strategy that decreases the intermediate data transfer between different datacentres and clouds. This helps to satisfy more tighter deadlines and reduce the cost of data transfer.

References

[1] Mell, P., Grance, T.: The NIST Definition of Cloud Computing. NIST special publication **800** (2011) 145

[2] Marz, N., Warren, J.: Big Data Principles and best practices of scalable realtime data systems. Manning Publications; First edition

[3] Dean, J., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clusters. In: Proceedings of the 6th conference on Symposium on Opearting Systems Design and Implementation (OSDI) 2004. OSDI'04, Berkeley, CA, USA, USENIX Association (2004) 10–10

[4] Lama, P., Zhou, X.: AROMA: Automated Resource Allocation and Configuration of MapReduce Environment in the Cloud. In: Proceedings of the 9th International Conference on Autonomic Computing (ICAC '12), ACM (2012)

[5] Hwang, E., Kim, K.H.: Minimizing Cost of Virtual Machines for Deadline-Constrained MapReduce Applications in the Cloud. In: Proceedings of ACM/IEEE 13th International Conference on Grid Computing (GRID'12). (2012)

[6] Lee, G., Tolia, N., Ranganathan, P., Katz, R.H.: Topology-Aware Resource Allocation for Data-Intensive Workloads. In: Proceedings of the First ACM Asia-Pacific Workshop on Systems (APSys'10). (2010)

[7] Mattess, M., Calheiros, R.N., Buyya, R.: Scaling mapreduce applications across hybrid clouds to meet soft deadlines. In: Proceedings of the 27th IEEE International Conference on Advanced Information Networking and Applications (AINA'13). (2013)

[8] Chvatal, V.: A greedy heuristic for the set-covering problem. Mathematics of operations research **4**(3) (1979) 233–235

[9] Buyya, R., Ranjan, R., Calheiros, R.: Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities. In: Proceedings of International Conference on High Performance Computing Simulation (HPCS'09). (2009)

[10] Kambatla, K., Pathak, A., Pucha, H.: Towards optimizing hadoop provisioning in the cloud. In: Proceedings of the First Workshop on Hot Topics in Cloud Computing (HotCloud '09). (2009)

[11] Verma, A., Cherkasova, L., Campbell, R.: Aria: automatic resource inference and allocation for mapreduce environments. In: Proceedings of the 8th ACM international conference on Autonomic computing (ICAC'11). (2011)

A Cloud Computing Service Level Agreement Framework with Negotiation and Secure Monitoring

V. Binu
Medies Software Labs
Bangalore
Email: binuv123@gmail.com

N. D. Gangadhar*
Computer Science and Engineering
M. S. Ramaiah University of Applied Sciences
Bangalore
Email: gangadhar.cs.et@msruas.ac.in

***Abstract*—Service Level Agreements (SLAs) are essential to the service oriented business model of Cloud Computing. SLA Frameworks for cloud computing need methodology for negotiation and monitoring of service delivery. The success and acceptability of SLAs requires their trusted and secure monitoring. In this paper an SLA Framework which incorporates negotiation and secure monitoring mechanism involving a third-party is developed. A reference implementation of the Framework is carried out and used to develop a case study of video transcoding service by a provider hosted on a cloud. The case study illustrates the effectiveness of the developed framework in creating SLAs for complex scenarios encountered in cloud computing services. Workload characterization is used to characterise the SLA service parameters which are employed in defining services.**

***Index Terms*—SLA, Cloud Computing, SLA Monitoring, SLA Negotiation, Video Transcoding, Security, Workload Characterization**

I. Introduction

Cloud Computing has emerged as a dominant model for service oriented computing system development [1]. The elasticity in service and the compelling business model it allows are the main reasons for this emergence. Such a service oriented business model would require Service Level Agreements (SLAs). There has been quite a lot of research effort in defining SLAs for cloud computing in the last few years; yet a lot of challenges remain [2]. However, very few actual implementations has emerged and this lack of SLA support is found to be a impedement in the present cloud systems [3]. In deed, even major comercial cloud providers have only static and gross SLAs [4].

This paper reports the design, reference implementation and a use case development of an SLA for cloud computing systems. It introduces secure monitoring of the SLA via the participation of a third party. While securing the custormer data on the cloud has received wide attention [4], [5], securing the SLA monitoring has not been addressed in literature.

The SLA framework is designed in such a way that it can be used by the service providing organization to configure and build specific services and deploy it for the consumer to form an agreement. The framework also acts as a platform for the consumer to negotiate the SLA parameters with the provider before forming the agreement. This forms the QoS with respect to the service and can be monitored by a trusted third party during the operation.

The SLA Framework development has followed the general theme of the WSLA [6] which was originally developed for Web Services. Certain aspects are simplied here for sake of implementation while at the same time certain aspects are extended.

An SLA will be formed between the cloud provider and the consumer before using the services. For any services, SLA parameters are identified by the provider which can be negotiated by the consumer. These SLA parameters are the quality of service parameters. However, from the service provider perspective, these have to be mapped to workloads. We envisage that service providers perform some form of workload characterization before defining a the service. Customers with the required technical expertise and resources, such as business entities, also would perform workload characterizations. Once the workload characterization is carried out, an SLA template will be built by the provider of the service using the SLA Framework and published. The consumer intending to use the service accesses the template and fills in the necessary information like consumer details, performs negotiation on the SLA parameter and sends it to the provider. The provider verifies the details, especially the SLA parameters, sent by the consumer and if it can meet those parameters, generates an SLA. Else, the service provider will ask for re-negotiation.

The Framework supports third party monitoring. A Third Party or Supporting Party, trusted by both the provider and the consumer, monitors and evaluates the service in accordance with the SLA. The consumer selects a mutually trusted support party while forming the SLA with the provider. The Supporting Party will have the details the SLA parameters agreed between the Signatory Parties and the action that needs to be carried out during any violation of those parameters by the provider. During the operation, a secure monitoring service periodically sends the SLA parameter measurements that must be guaranteed to the supporting party for evaluation. The supporting party verifies the metric based on the measured values received from the provider against that in the agreement. If the computed metric is found to be out of the range, it will carry out the action as stated in the document.

The rest of the paper is organised as follows. Section II describes the Framework architecture, its components, formation

*Corresponding Author

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

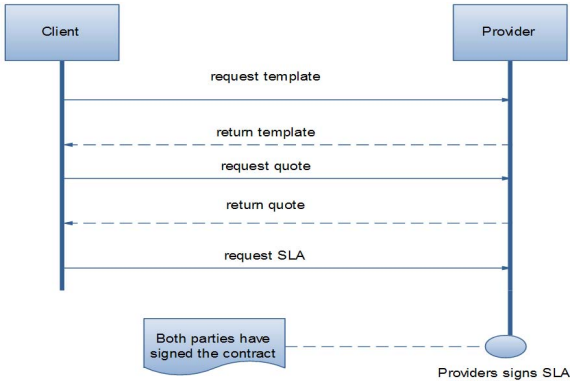


Fig. 1. Service Level Agreement Process

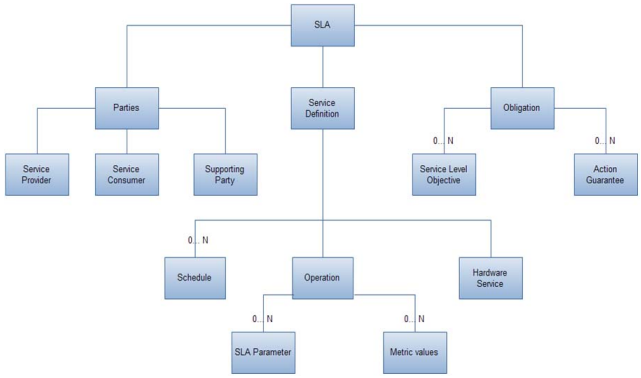


Fig. 2. SLA Document Structure

and run time management. Secure monitoring for SLA with third party involvement is discussed in Section III. Section IV provides a high level view of a reference implementation of the Framework with low level details relegated to Appendix A. A case study of cloud hosted video transcoding service developed to demonstrate the use of the Framework in complex scenarios is presented in Section V. Section VI concludes the paper.

II. The SLA Framework

The SLA framework is designed to include formation of service definition and SLA document generation, run time management and the role of third party.

Figure 1 depicts the Service Level Agreement process. The the steps involved in the SLA are:

1) The provider builds a SLA template using the framework that consists of provider details, SLA parameter names, service level agreement and guaranteed actions
2) The client fetches the template and fills the information about the client and the required metric values for every SLA parameter provided by the provider
3) The client after filling the form, sends it for approval
4) If the provider feels that it can satisfy the consumer requirement, then an acknowledgement will be sent
5) An SLA document will be generated that identifies different services accessed by different users

The SLA document is divided into three main components

1) Parties
2) Service Definition
3) Obligation

Figure 2 depicts the document structure and its components.

a) Parties: This component indicates the parties who are involved in the SLA process. These parties include:

Service Provider Forms the SLA for a particular service for its consumers and provides details of SLA parameters for the service. It also indicates the action interfaces that are exposed to other parties like service consumer and supporting party.

Service Consumer The one who wants to use the service will form an SLA with the provider by providing the necessary SLA parameter metric values and also details of the consumer.

Supporting Party Performs monitoring and notifies the signatory party during SLA violation. If violated, sends notifications as defined by the action guarantee.

b) Service Definition: Indicates the service name and the type of service that the provider provides. The sub-components of service definitions are:

Schedule Describes the availability of the service for its consumer. It also specifies that the service will be guaranteed during the schedule for which consumer selects.

SLA Parameter These parameters are specific to particular services. The parameters form the main aspect of differentiating the consumers on the charges. The negotiation will be done based on these parameters

Metric defines the values for the SLA parameters which are given by the consumer for their service.

Hardware Service Indicates the hardware resource on which the SLA is performed and should be guaranteed during the service.

c) Obligation: Defines services that are guaranteed. It also states the action that needs to taken if the SLA is violated. The sub components are:

Service Level Objective The SLO contains description of the SLA parameters that are guaranteed during the operation of the service.

Action Guarantee specifies the appropriate action that needs to be executed if the guaranteed SLA mentioned in service level objective is violated.

A. SLA Formation and Service Definition

The SLA is formed with respect to a particular service by the provider. The service description is described using WSDL defining the interface between the service and the application using it. The SLA parameters for any service are formed by identifying the key parameters that determine the performance of its operation. The metric values are calculated and are evaluated before forming the SLA parameters for the service. The client could also characterize the metric values in order to choose an optimum service level. The service provider also runs several workload characterizations and characterizes parameters such that it will provide maximum performance for their published service as indicated in the Figure 3.

The metrics derived can be monitored either at the server side or at the client side. For guarantees, a mutually agreed third party provides the metric value measurement during the operation. The way of accessing these parameters is defined

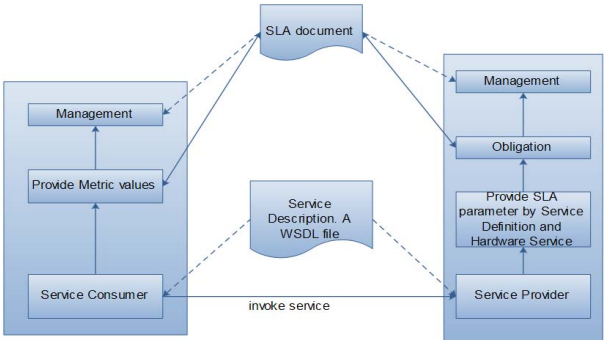


Fig. 3. SLA Formation and Service Definition

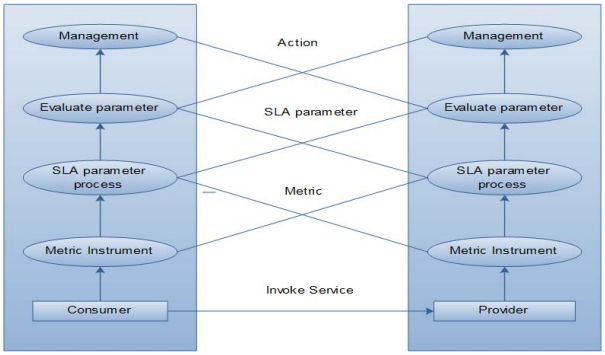


Fig. 4. Run Time SLA Management

in the SLA document agreed between the consumer and the provider.

The relationship between the SLA document and the management system is dynamic in nature. Management activity will be invoked based on the obligation defined in the document. Many service level objectives can be declared and defined for which services are guaranteed. If any violations of these service parameters are found, then the corresponding action mentioned in the document will be executed and are reported to the management system of consumer and the provider.

B. Run Time Management of SLA

The measurement and management functionalities contribute to the run time management of the SLA as illustrated in Figure 4. It also depicts the parties interacting at various levels for the agreed SLA.

The measurement functionality includes the metric value for the SLA parameter and the service level objective indicating the SLA parameters that must be guaranteed. The metric value calculated must provide an optimum system performance and can be negotiated during the formation of an agreement.

The set of metrics that are used to guarantee services in the SLA document are made available as management functions called as SLA parameters. These parameters are evaluated during the service and corresponding management functions are called whenever any violation of SLA parameters occurs.

The Condition Evaluation function evaluates the SLA parameters that are guaranteed. These guarantees are defined by the "predicate" field of the SLA document. The values that are guaranteed are defined by the SLA parameter metrics referred

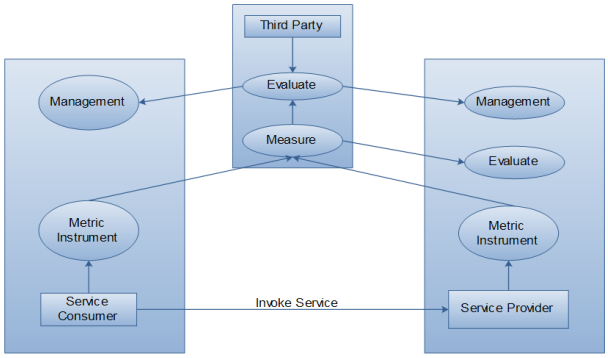


Fig. 5. Role of Thirdy Party in SLA Process

to by the management functions. The condition evaluation function will verify the Predicate field as the service proceeds. If the predicte is found to be true, then it results in violation and corresponding action will be executed.

The management functions are invoked when there is any violation in the SLA. The Action Guarantee suggests the required measure to be taken for managing the SLA violation. It might be a simple notification messages or any other, potentially complicated, actions as specified in the SLA document.

III. Secure Monitoring with Third Party Participation

In a distributed web environment, provider and consumer may commission a third party to validate the results of any operations. These parties are called as supporting parties. They are defined while forming the SLA document. The supporting parties are either sponsored by the provider or consumer or both. The role of supporting party is indicated in Figure 5.

The supporting party will perform following roles

- They perform measurement functions in which the SLA parameters metric values are made available to the provider for a consumer who banks on the supporting party.
- The condition evaluation activity is performed by supporting party whenever there is any violation in the SLA parameter that is guaranteed.
- Metric Instrument: Based on the condition evaluation corresponding management actions are performed that are defined in the SLA document

In order to make the monitoring process secure (and hence enhancing the trust) the monitoring service data is secured using symmetric key cryptography.

IV. Framework Implementation

An implementation of the framework is carried out in Java. This is used to develop a case study discussed in Section V. The developed Java classes and their corresponding member functions are made available to the users of the framework to develop applications in order provide services.

```
public class SLA {
  Parties parties;
  ServiceDefinition servicedefinition;
  Obligations obligations;
  public SLA() {
    parties = new Parties(); // instantiate Parties
```

```
    servicedefinition = new ServiceDefinition();
      //instantiate ServiceDefinition
    obligations = new Obligations(); //instantiate Obligations
  }
}
```

The SLA document structure is defined as an XML document:

```
<? xml version=”1.0”>
<SLA xmlns=”http://localhost:8080/SLAFrameworkService/
    sla_video_transcode.xml”
  xmlns:xsi=http://www.w3.org/2001/XMLSchema−instance
  xsi:schemaLocation=”http://localhost:8080/
    SLAFrameworkService/sla.xsd”>
<Parties>
...
</Parties>
<ServiceDefinition>
...
</ServiceDefinition>
<Obligations>
...
</Obligations>
</SLA>
```

The equivalent schema definition of the document that defines the namespace and types:

```
<xs: element name=”SLA”>
  <xs: complexType>
    <xs: sequence>
    <xs: element name=”Parties”
      type=”sla:PartyType”>
      <xs: complexType>
      <xs: sequence>
        ...
      <xs:elementname=”ServiceDefinition”
        type=”sla:ServiceDefinitionType”>
      <xs: complexType>
        <xs: sequence>
          ...
        <xs: element name=”Obligations”
          type = ”sla:ObligationsType”>
      <xs: complexType>
        <xs: sequence>
          ...
        </xs: sequence>
  ...
  </xs: complexType>
</xs: element>
```

The element SLA forms the root element of an SLA document. It contains sections like parties, service definition and obligations. To maintain a common convention all the elements of the document are given the type as “sla”.

Further details of the SLA Framework implementation can be found in Appendix A.

V. Case Study: Video Transcoding Service

To demonstrate the use and effectiveness of the developed SLA framework, a video transcoding service hosted by the service provider on a Cloud service provider is developed. The Computing Cloud provides the computing resource like CPU cycles and performs video transcoding service for the hosted service provider. The transcoding service is used to provide video content in multiple formats to the end users suiting their different devices and media players.

Typical workload parameters for video transcoding service are video resolution, frame per second, video format and bit rate. The time needed to transcode any video file depends on CPU usage and the workload parameters. The consumers (end users) are classified and charged based on their CPU usage.

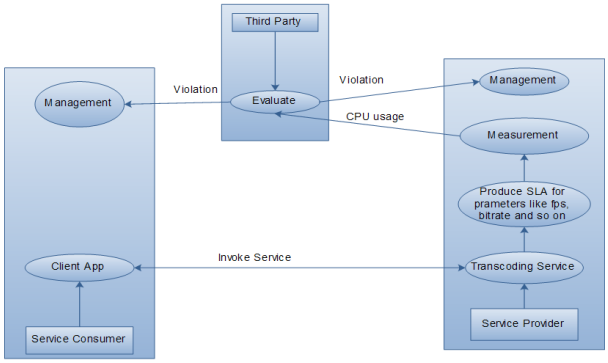


Fig. 7. Video Transcoding Service Using the Developed SLA Framework

Thus the SLA parameters for the video transcoding service are fps, resolution, bitrate, format and CPU usage which can be negotiated by the consumers while forming the SLA.

Figure 6 depicts the overall scenario of the video transcoding service. SLA1 is the SLA formed between the transcoding service provider offering the transcoding service and the consumer who uses the service. The SLA parameters are negotiated while forming the agreement. Once the agreement is done, the consumer can upload video files for transcoding according to the SLA. Consumer notifies the Third Party1 (between the provider and the consumer) as it uploads the file. The Third Party1 contains the details of the SLA parameters and action that needs to be carried out during any violations. The provider verifies the video parameters and starts transcoding and periodically sends SLA parameter value to the Third Party1 for evaluation. Both the Third Parties evaluate SLA parameter values in order to identify any violation. If any violation happens, then the SLA document will be referred and corresponding actions will be carried out.

Another SLA, indicated as SLA2 in Figure 6, will be formed between the service provider and the cloud and it will be evaluated and verified by Third Party2.

The SLA Framework classes and associated methods are extensively used in building the transcoding service that incorporates negotiation, supports third party monitoring and guarantee services in accordance with the agreement.

A. Design of the Video Transcoding Service

Figure 7 depicts the employment of the SLA Framework and the video transcoding application illustrating how the framework can be used to provide a transcoding service and maintain SLA throughout the processing.

The SLA in Figure 7 illustrates the relationship between the transcoding service provided by a service provide and a consumer. The provider offers the transcoding service method getTranscodingService() through SOAP over HTTP. The overall description of the transcoding service will be found in a WSDL file that will be published by the service provider for its consumers. The consumer and the provider form an SLA for the video transcoding parameters such as frame per seconds, bit-rate, resolution and CPU usage. These parameters will be negotiated and the negotiation activity is part of the framework activity itself.

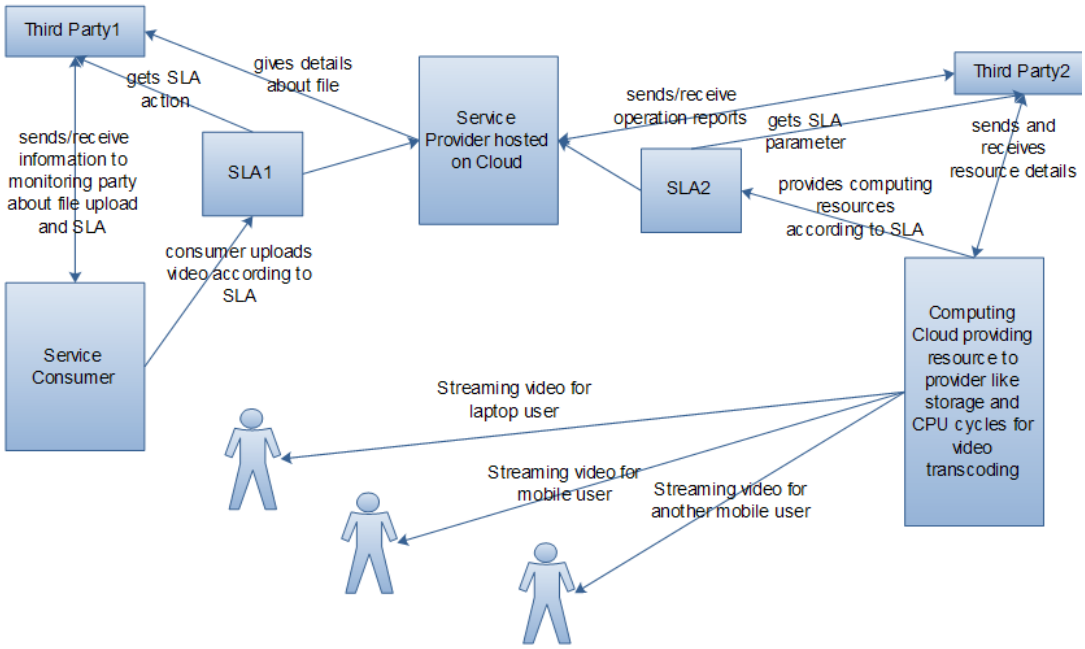


Fig. 6. Video Transcoding Service Architecture

Once the agreement is done using the framework provided methods, an SLA document will be generated that involves the details of the agreement like parties' details, service definition parameters and obligations. The provider will send the details of the agreement to consumer and even to the third party. Usually the details sent to the third party will include the SLA parameters and the corresponding action to be done if the SLA is violated. During the transcoding service for any consumer video file, the measured data will be sent from the service provider to the third party. The third party runs an evaluation function to figure out whether the SLA parameters are satisfied. If there is any violation, it informs the management function defined that will respond according to the action defined in the SLA document.

B. Implementation of the Video Transcoding Service

The service provider is implemented making extensive use of the developed SLA Framework classes (*vide* Appendix A) to build the SLA template. The template contains the provider details, SLA parameters of the service that needs to be negotiated, schedule of the service and the obligations like Service Level Objective and Action Guarantee. In this implementation, JSPs are developed as an interface to call the classes of the SLA framework. The framework classes are invoked as the JSP fields are populated with the values that are necessary to form the template by the provider:

```
/*Defining the Parties section of the SLA*/
  String pname = request.getParameter("pname");
  String pstreet = request.getParameter("pstreet");
  String pcity = request.getParameter("pcity");
  String pacttype = request.getParameter("pacttype");
    // also provides the details of the WSDL file and action parameters
  String pactname = request.getParameter("pactname");
...
```

Internally, the method calls are to the SLA framework class, as in the following

```
sla.parties.serviceprovider.setName(temp[0]);
// calling SLA class Parties to set service provider name.
String provider = sla.parties.serviceprovider.getName();
// retrieving the parameter set to build the SLA template
sla.parties.serviceprovider.contact.setStreet(temp[1]);
String prstreet = sla.parties.serviceprovider.contact.getStreet();
```

The complete template is constructed by calling the SLA framework classes similarly and is published for consumers to form an SLA before accessing the service.

```
<ServiceProvider name="MSRSAS Video Transcoding Service">
  <Contact>
  <Street>Peenya 2nd Stage</Street>
  <City>Bangalore</City>
  </Contact>
  <Action type="MSRSASSOAPOperationType" name="Notification"
      partyName="MonitoringParty">
    <WSDLFile>Transcode.wsdl</WSDLFile>
    <SOAPBindingName>TranscodeSoapBinding
      </SOAPBindingName>
    <SOAPOperationName>videotranscode</SOAPOperationName>
  </Action>
</ServiceProvider>
```

The WSDL file specifies the details of the service location and its availability. The action name "Notification" states that the monitoring party (support party) should provide notification service during the violation of SLA. The template containing the SLA parameters is as shown below; the SLA parameter represents for "flv" format.

```
<SLAParameter name="FramePerSecondFlv" type="int" unit="fps">
  <Metric>FramePerSecond</Metric>
<SLAParameter name="CPUusage" type="int" unit="" ">
  <Metric>CPUusage</Metric>
```

Once the SLA is agreed, the client uploads the video file to be transcoded. The provider starts transcoding by getting the parameters from respective consumer SLA document with the help of SLA framework classes and transcodes accordingly.

```
pm.getParseDetails(fname);
// call the parse function for the consumer (fname is the
// associated consumer SLA document). The parse function
```

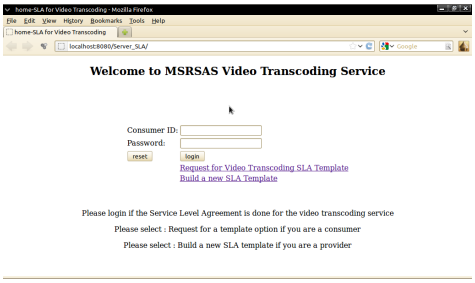


Fig. 8. Transcoding Service Start Page

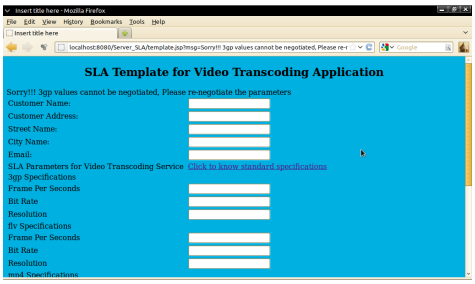


Fig. 9. SLA Negotiation

```
// internally calls the framework classes
metricparser = new Metric(); //calling the Metric class of SLA framework
metricparser.setName(attributes.getValue("name"));
metricparser.setType(attributes.getValue("type"));
```

During this process, the CPU usage mentioned for the consumer will be identified and the CPU resources are controlled by a controlling application. During the operation, the monitoring application generates an encrypted log report that contains the processor usage and sends it to the support party periodically. The support party receives the encrypted data from the provider, decrypts it and performs an evaluation activity by referring to the data present in the document. If there is any violation i.e., if the provider is unable to satisfy the consumer with the promised SLA parameter, then the corresponding action will be taken as mentioned in the action field of the document. Typically, a "Notification" activity is mentioned in the document.

C. Results

Figures 8–11 depict the resulting screenshots from various stages of the SLA process for the implemented Video Trancoding Service case study. All the aspects of the SLA Framework have been tested and the results validate the SLA process from formation through negotiation and renegotation to monitoring.

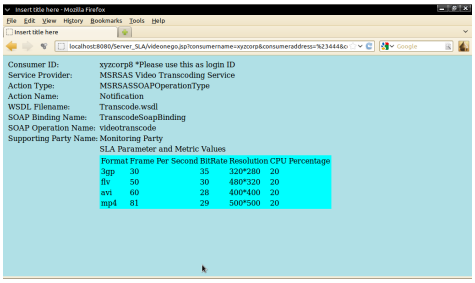


Fig. 10. Consumer Acknowledgment

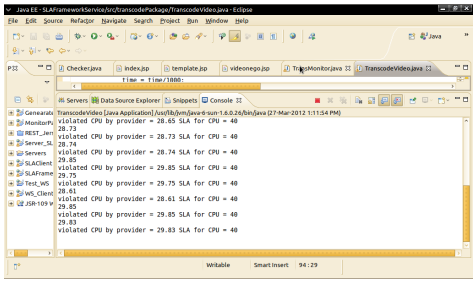


Fig. 11. SLA Monitoring

VI. Conclusion

A SLA Framework for cloud computing systems is designed and implemented. The Framework incorporates negotiation as well as secure monitoring of the SLA parameters. The Framework is illustrated using the case study of a video transcoding service hosted on a cloud. This typical cloud application scenario has demonstrated the versatality of the developed framework. Trust management can be added to the security model to enhance the usefulness of the Framework.

References

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility," *Future Generation Computer Systems*, vol. 25, pp. 599–616, 2009.

[2] R. Buyya, S. K. Garg, and R. N. Calheiros, "SLA-Oriented Resource Provisioning for Cloud Computing: Challenges, Architecture, and Solutions," in *2011 International Conference on Cloud and Service Computing*. IEEE, 2011.

[3] W.-T. Tsai, X. Sun, and J. Balasooriya, "Service-Oriented Cloud Computing Architecture," in *2010 Seventh International Conference on Information Technology*. IEEE, 2010, pp. 684–689.

[4] M. Al Morsy, J. Grundy, and I. Mueller, "An Analysis of The Cloud Computing Security Problem," in *Proceedings of APSEC 2010 Cloud Workshop*, 2010.

[5] S. Subashini and V. Kavitha, "A Survey on Security Issues in Service Delivery Models of Cloud Computing," *Journal of Network and Computer Applications*, vol. 34, pp. 1–11, 2011.

[6] H. Ludwig, A. Keller, A. Dan, R. P. King, and R. Frank, "Web Service Level Agreement (WSLA) Language Specification," Specification, 2003.

Appendix A
Framework Implementation Details

This Appendix provides the details of the SLA Framework implementation.

a) Parties: The Parties section is a complex definition that includes the details of the service provider, service consumer and the supporting party. The combination of both service provider and service consumers are called as signatory parties who form the agreement for the service. Supporting party provides monitoring function of the service. The complex type "Parties" schema definition is defined as follows

```
<xs: element name="Parties" type="sla:PartyType">
  <xs: complexType>
  <xs: sequence>
  <xs: element name="ServiceProvider"
    type="sla:ServiceProviderType">
  <xs: complexType>
  ...
  <xs: element name="ServiceConsumer"
    type="sla: ServiceConsumerType">
  <xs: complexType>
  <xs: element name="SupportingParty"
    type="sla:SupportingPartyType">
```

The “ServiceProvider” is the party that provides the service and guarantees. The “ServiceConsumer” is the recipient of the service. The “SupportingParties” is involved in measuring the service’s parameters, supervising the given guarantees and managing the corrective procedures in case of failure. The other fields that come under ServiceProvider, ServiceConsumer and SupportingParties are “Contact” that defines the address and respective parties contact details and “Action” that is invoked when any violation occurs in the SLA parameters. Typically a WSDL file will be mentioned that accepts the value does the corresponding action.

```
<xs: element name=“Contact”>
...
<xs: element name=“Action”>
<xs: complexType>
<xs: sequence>
<xs: element name=“WSDLFile” type=“xs: string”/>
...
```

The follwoing Java class provides the Party definition.

```
public class Parties {
  ServiceProvider serviceprovider;
  ServiceConsumer serviceconsumer;
  SupportingParty supportingparty;
  public Parties() {
    serviceprovider = ServiceProvider(); //initialize provider
    serviceconsumer = ServiceConsumer(); //initialize consumer
    supportingparty = SupportingParty(); //initialize support party
  }
}
```

The framework class that is responsible for the creation of the service provider details is defined as

```
public class ServiceProvider {
  String name;
  Contact contact;
  Action action;
  public ServiceProvider() {
    contact = new Contact(); //instantiate contact
    action = new Action(); //instantiate action
    /* Setters and getters method that are used set and
    get the value respectively */
    public void setName(String name) {
      this.name = name; // set the name of service provider
    }
    public String getName() {
      return name; // get the name of provider
    }
  }
}
```

Contact and Action are the other two classes that the “ServiceProvider” class instantiates in order to set the values for the fields that are a part of “ServiceProvider”. The “ServiceConsumer” and “SupportingParty” also has the similar kind of class definition and does instantiate “Action” and “Contact” classes.

b) Service Definition: This section of the SLA document contains the detailed specification of the service. It includes the references to the service operation as well as the information needed to define service level guarantees. The other subsections of the Service Definition include

- Schedule
- Operation

The “ServiceDefintion” section is defined as follows in the framework.

```
<xs: element name=“ServiceDefinition”
    type=“sla: ServiceDefinitionType”>
  <xs: complexType>
  <xs: sequence>
  <xs: element name=“Schedule” type=“sla: ScheduleType”>
  <xs: complexType>
  ...
  <xs: element name=“Operation” type=“sla: OperationType”>
  <xs: complexType>
  ...
  <xs: element name=“HardwareService”
    type=“sla: HardwareServiceType”>
  <xs: complexType>
  ...
  </xs:sequence>
  </xs:complexType>
</xs:element>
```

The “Schedule” subsection indicates the service schedule: the provider can schedule the service availability which defines the time at which the service is provided. This enables the consumer to know the time at which the service is available. The “Period” part of “Schedules” defines the validity of the service. Every consumer who forms an SLA with the provider for a service will be given the validity of the SLA. The validity of the SLA expires once the period crosses the end date. “Interval” suggests the time interval (in minutes, for instance) for which service will be scheduled.

“Operation” is again a complex type that will be described later. It indicates the type of operation that a service provides and its name. The “HardwareService” mentions the hardware resources of the system on which the service is executed has to be monitored. Typically, the resources will be system resources (e.g., memory and CPU usages) that are consumed by any service running on a system. This makes the framework flexible so that it can be used to define resources of the system also. The Framework’s class that defines and sets “Service Definition” and its subsections are given below.

```
public class ServiceDefinition {
  String name;
  Schedule schedule;
  Operation operation;
  HardwareService hardwareservice;
  public ServiceDefinition() {
    schedule = new Schedule(); //create an instance for schedule
    operation = new Operation(); //create an instance for operation
    hardwareservice = new HardwareService();
      //create an instance for hardware
    }
  }
}
```

The class for “Schedule” that enables the definition of the schedule parameter is

```
public class Schedule {
  String name;
  Period period;
  Interval interval;
  public Schedule() {
    period = new Period(); //Schedule period
    interval = new Interval(); //Schedule interval
    ...
```

The “Operation” section of the “ServiceDefinition” contains the SLA parameter names and their corresponding metric values defined by the “Metric” class of the framework.

```
<xs: element name=“Operation” type=“sla: OperationType”>
<xs: complexType>
  <xs: sequence>
  <xs: element name=“SLAParameter” type=“sla: SLAParameterType”>
```

```
<xs: complexType>
...
<xs: element name="Metric" type="sla: MetricType">
<xs: complexType>
...
```

The framework class that is responsible for initiating the "Operation" and its associated subsections "SLAParameter" and "Metric" is

```
public class Operation {
  String name, type;
  SLAParameter slaparameter;
  Metric metric;
  public Operation() {
    slaparameter = new SLAParameter(); //instantiate SLAParameter
    metric = new Metric(); // instantiate Metric
  ...
```

The definition for "SLAParameter" and "Metric" are as follows:

```
public class SLAParameter {
  String name, type, unit, metric;
  Communication communication;
  public SLAParameter() {
    communication = new Communication();
      //setting up communication to indicate the party
      //responsible for setting SLA parameter
    }
    public class Metric { // class for metric definition
      // that sets value for the SLA Parameter
      String name, type, unit, source, schedule,value;
      public Metric() {
      }
...
```

The metric defines the value for the SLA parameters for the service. The consumer may accept the metric values provided by the service provider or the consumer can negotiate for the metric values for the parameters.

c) Obligations: The "Obligations" section of SLA document defines the parameters that are guaranteed. It also mentions the action that needs to be performed if the parameters promised are violated. The two main sub section of the "Obligations" are "ServiceLevelObjective" (SLO) and "ActionGuarantee" (AG).

```
<xs: element name = "Obligation" type = âĂIJsla: ObligationTypeâĂİ>
<xs: complexType>
  <xs: sequence>
  <xs: element name = "ServiceLevelObjective"
    type = "ServiceLevelObjectiveType">
    <xs: complexType>
    ...
    <xs: element name = "ActionGuarantee"
      type="ActionGuaranteeType">
    <xs: complexType>
      ...
```

The class that defines obligations is

```
  public class Obligations {
    ServiceLevelObjective sevicelevelobjective;
    ActionGuarantee actionguarantee;
    public Obligations() {
      sevicelevelobjective =
        new ServiceLevelObjective(); //instantiate SLO
      actionguarantee = new ActionGuarantee(); //instantiate AG
      ...
```

d) Service Level Objective: A Service Level Objective mentions a service that will be guaranteed to be served in a given period of time. It usually describes the SLAParameter that needs to be guaranteed.

```
<xs: element name="ServiceLevelObjective"
  type="sla: ServiceLevelObjectiveType">
<xs: complexType>
<xs: element name="Obliged" type="xs: string"/>
<xs: element name="Validity" type=âĂIJsla: ValidityTypeâĂİ >
...
```

The Obliged element is a reference to a party that is in charge of delivering what is promised in this obligation. The Validity element in the "ServiceLevelObjective" defines the validity for the guaranteed service An Expression defines the actual content of the obligation, that is, what is asserted by the service provider to the service customer. An EvaluationEvent defines the case in which the expression of the service level objective is to be evaluated.

A Predicate element is a Boolean definition in which the SLAParameter minimum or maximum value will be declared using value tag. If the SLAParameter turns out be less than the value indicated in the "ServiceLevelObjective" and the Predicate is stated as less, then the Predicate condition becomes true, which concludes that the SLAParameter could not be guaranteed.

The equivalent class of the framework implementation is

```
public class ServiceLevelObjective {
  String name, obliged;
  Period validity;
  Expression expression;
  public ServiceLevelObjective() {
    validity = new Period();
    //instantiate to define the validity of the SLO
    expression = new Expression();
    //used to define the Predicate and SLA value
  }
}
```

e) Action Guarantee: If the "ServiceLevelObjective" could not be served by the provider which can be identified by the result of Predicate element then corresponding action will be taken as defined in the ActionGuarantee. The ActionGuarantee will include the supporting party who monitors the service.

```
<xs: element name = "ActionGuarantee" type="sla: ActionGuaranteeType">
<xs: complexType>
<xs: sequence>
...
```

The framework class definition for the element ActionGuarantee is as follows:

```
public class ActionGuarantee {
  String name, obliged, evaluation_event,
  execution_modality;
  Expression expression;
  QualifiedAction qualifiedaction;
  public ActionGuarantee() {
    expression = new Expression();
    //instantiate expression
    qualifiedaction = new QualifiedAction();
    //instantiate qualified action
  }
  ...
```

Obliged is a reference to a party that is in charge of delivering what is promised in this guarantee. Expression defines the pre condition of the action. An action level guarantee may have an EvaluationEvent associated with it. It defines the case in which the expression of the service level guarantee is to be evaluated. The QualifiedAction contains a definition of the action to be performed at a particular party.

A HYBRID PROTOCOL TO SECURE THE CLOUD FROM INSIDER THREATS

Sriram M, Vaibhav Patel, Harishma D, Nachammai Lakshmanan

Department of Computer Science and Engineering
National Institute of Technology, Tiruchirapalli-India

ABSTRACT

Data Outsourcing has evolved rapidly with the advent of cloud computing wherein third parties provide storage services. Insider attacks still continue to haunt cloud users as they tend to cause unprecedented damage, especially when privileged users who have access to sensitive information go rogue. Many proposals have been made to Secure the Cloud from Insider threat Attacks and most of the standard approaches have been proven to fail from time to time. In this paper, an implementation of a Hybrid protocol that uses Selective Encryption with data cleaning, Enhanced Neural Network based user profiling and decoy technology to combat the insider threat has been proposed. The proposed system gave unprecedented level of security.

Index Terms— Insider Threat Attacks, Selective Encryption, Data Cleaning, Decoy Technology, Neural Networks, User Profiling

1. INTRODUCTION

Cloud Computing is a ubiquitous phenomenon today. We see a number of companies accelerating towards cloud adoption with the assumption that their data is safe and secure. However, many of them still have apprehensions about putting their software on someone else's hardware for obvious security related issues such as data loss, phishing etc. With the advent of the multi-tenancy model in cloud computing new security threats have been introduced [1]. There is an added threat of a malicious insider misusing information that he is not authorized to use leading to data thefts. [2] Concludes that a majority of insider attacks were only detected when there was a noticeable irregularity in the information system. Researchers [3] have shown on how easy is it for an insider to get access to customer admin passwords and access sensitive information without the knowledge of the customer. Modern malware attacks use obfuscation techniques to circumvent security controls on the network and resemble normal user access patterns and network traffic. While advanced malware has certainly garnered a lot of attention, insider threats (i.e., threats posed by employees, third parties, or malicious software that uses legitimate access rights to networks, applications, and sensitive data as an attack vector) also present a number of challenges. Cloud security researchers have examined this insider attack in detail and come up with sophisticated encryption algorithms, none of them proven to be a sufficient data protection algorithm when used alone.

2. EXISTING SOLUTIONS

The current generation of cloud computing infrastructures does not provide security against untrustworthy cloud operators making them unsuitable for storing sensitive information. To address this issue various projects are being pursued that range from theory to practice. With this respect, a plethora of encryption solutions for Insider threat attacks that include Homomorphic Encryption were proposed. Encryption provides confidentiality by hiding all useful information about the plaintext. However, this renders data useless, in the sense that one loses the ability to operate on it. A lot of algorithms have been designed to operate on encrypted data but then these operations are very costly. There is a possibility of an attack, if the user of the cloud decrypts the data locally on the cloud and does not delete it after use.

[4] Proof of Storage algorithms (also known as a proof of data possession or a proof of irretrievability) are designed so as a client can verify whether the cloud operator has tampered with its data.

[5]Fog Computing refers to the creation of bogus, "decoy" information placed in the cloud along with otherwise true information to hide what is true from what is bogus. This strategy provides a "fog" of misinformation to protect sensitive, real information in the cloud. Users of this concept need a number of "decoy documents" as bait, to be placed along with their actual data to detect and defend against insiders who may gain illegitimate access to a host and steal information. The insider may be human, or a Trojan malware program planted to steal credit cards, or logins to online banking or other credentials from the user. While this approach gives robust results, it does not guarantee that the system is free from insider threats as a lot depends upon chance and probability.

We have seen how algorithms or protocols that rely only on complex encryption algorithms compromise largely on efficiency. It is fair to say that these algorithms have time

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

and again failed due to a plethora of reasons such as buggy code, sophisticated attacks, malicious insiders etc. [6]

In this paper, a new hybrid protocol to secure the cloud from Insider attacks has been proposed. It leverages on the advantages of encryption and fog computing. It uses an approach called selective encryption, wherein only selected information (as decided by the cloud user based on his discretion) that requires maximum security is encrypted. To protect the data from insider attacks in the name of maintenance (when the user locally decrypts data), an approach known as data cleaning is used. Since, this process would require certain efficiency compromises, for the remaining not-so sensitive data decoy technology and user profiling has been used [5].

The rest of the paper is organized as follows. SECTION 3 focuses on the key principles, SECTION 4 explains the Algorithm, SECTION 5 gives the implementation details and SECTION 6 the conclusion.

3. KEY PRINCIPLES

3.1 Selective Encryption

To completely guarantee that the system is safe and secure, the cloud user in the proposed system is given an option to store all sensitive data in an encrypted format. Immediately post encryption the key shall be deleted from the system. Since the key has been deleted, a malicious user will not be able to see the data in a decrypted format, serving the purpose.

But this approach has its own pitfalls such as:-

1. All data cannot be encrypted because of performance issues.

2. When a legitimate user decrypts the data, the decrypted data would be stored in the physical machine in its volatile memory for a temporary period, during which the malicious insider could steal/misuse the sensitive information.

To address the first concern, the user is given an option to completely encrypt/ selectively encrypt/ not encrypt his data. The second concern has been addressed by a technique known as data cleaning. Data clean-up [7], as proposed here, implies that planned or unplanned access to the persistent or volatile memory is not possible until sensitive, critical or personal data has been deleted or reliably overwritten. This requires appropriate trigger signals, indicating to the data clean-up procedure, that planned access is requested, or unplanned access is imminent. Planned access postulates the creation of new trigger signals, whereas unplanned access can rely on perimeter security alarms as signals.

3.2 User Profiling

Legitimate users of a computer system are familiar with the file/directory structures. Any search for specific files is likely to be targeted and limited. A masquerader, however, who gets access to the victim's system illegitimately, is unlikely to be familiar with the structure and contents of the file system. Their search is likely to be widespread and untargeted. Based on this key assumption, we profiled user search behavior and developed classifiers based on Neural Networks.

The Neural Network in our implementation took into account the following parameters:-

Volume of Data Downloaded: Based on the data download threshold violations, alerts are generated. This parameter is considered because the masquerader would tend to download multiple files as he is not expected to understand the directory structure.

Nature of operations: Consider the example of Querying. DB administrators and System administrators usually query large data. If for some reason the size of the query is small the user could be a potential masquerader.

Division of Task: Business Analysts would not be concerned with Development Activities/Source Code, Developers would not be interested in financial data, DB administrators would not be interested in the application, Marketing folks would not be interested in System/DB performance etc. Clearly data accessed by users would be related and restricted to their domain. If someone goes across domains to access data, he is a potential masquerader.

IP/Locale: The IP address, Locale Time, Time Zone etc are used to determine if the user is a legitimate user. If there is a violation in the usual patters, the user is a potential masquerader.

Log File Contents for determining User behavior: The log files provide important details such as clicks made by the user while visiting a website. If the user opens a particular website and does some other work outside the system then it may be considered as the usage of the website. The details related to clicks made by the user and the time to scroll etc was noted for effective data mining. The situation where a user does some operation on the cloud and simultaneously opens another website in another browser or opens two browsers at the same time to perform some operation has also been taken into account. By making note of the time stamps in the log, and relevant entries regarding these subtle operations we determine if the user is an insider or a legitimate user

Based on the application, several other strategies could be used to determine if the user is malicious or a not.

3.3 Decoys

Fog Computing refers to the creation of bogus, "decoy" information placed in the cloud along with otherwise true information to hide what is true from what is bogus. This strategy provides a "fog" of misinformation to protect sensitive, real information in the cloud. In this paper we implement an algorithm where we use machine generated data called decoys which generate email alerts whenever they are touched by an attacker and generate false data to

confuse and confound the insider on what is real and what is not. We use decoy documents to store these false information and provoke an attacker to steal this document. Decoys have a number of properties. They should be:

1. believable (so an attacker will consider the document to be real)
2. conspicuous (so an attacker can easily find the decoy)
3. enticing (so that the decoy is attractive and is likely to be opened by the attacker)
4. differentiable by the real user (so the real user knows what is real and what is not)
5. Non-interfering (so that the real user won't accidentally misuse the bogus information contained within the decoy).

3.3.1 Salient Features

An encryption of all selected data stored in persistent memory is assumed. In order to prevent the Encrypted data to be accessed by an insider in a decrypted format, the encryption key is deleted immediately after the encryption is done. Though the key is present only with the party that owns the information/business logic, the volatile memory could contain data in the decrypted format. Data clean-up, as proposed here, implies that planned or unplanned access to the persistent or volatile memory is not possible until sensitive, critical or personal data has been deleted or reliably overwritten.

All communication between the user and the cloud is assumed to be done over HTTP/HTTPS protocol. Each sever has a Master Agent through which the communication is done. Agent processes all incoming access requests, maintains a Queue and grants permissions to legitimate requests that do not cause any violations. Once a user has completed his critical tasks, it could either inform the agent asking it to trigger the Data Clean up procedure or the agent could continuously poll the Volatile memory and trigger Data Clean Up procedure whenever appropriate.

3.3.2 User Profiling

A user is classified as malicious or genuine based on the value of $h_\Theta(x)$. If its value is greater than 0.5 the user is Malicious, else genuine. $h_\Theta(x)$ is computed using a neural network as below. The parameters taken are Volume of Data Downloaded, Nature of Operations, Division of task, IP/Locale and Log File Contents for determining user behavior.

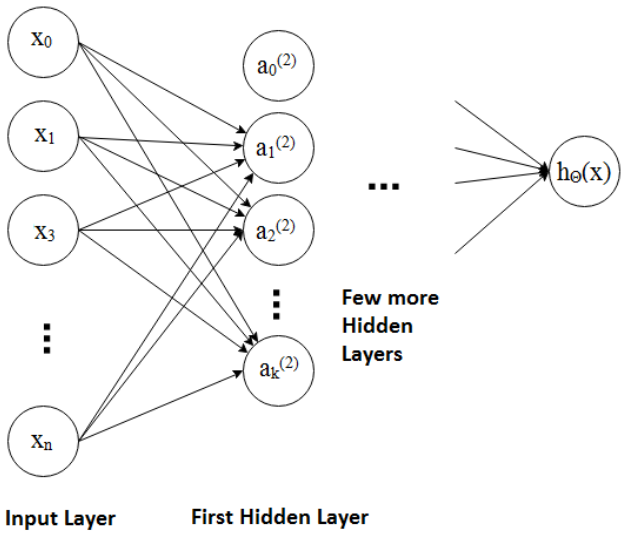


Fig.1. Schematic Representation of the Neural Network Implementation

The hypothesis $h_\Theta(x)$ is calculated according to the following equations:-

$$a_1^{(2)} = g(\Theta_{1.0}^{(1)}x_0 + \Theta_{1.1}^{(1)}x_1 + \ldots + \Theta_{1.n}^{(1)}x_n)$$
$$a_2^{(2)} = g(\Theta_{2.0}^{(1)}x_0 + \Theta_{2.1}^{(1)}x_1 + \ldots + \Theta_{2.n}^{(1)}x_n)$$
$$\ldots$$
$$a_{k(2)}^{(2)} = g(\Theta_{k(2).0}^{(1)}x_0 + \Theta_{k(2).1}^{(1)}x_1 + \ldots + \Theta_{k(2).n}^{(1)}x_n)$$

$$a_1^{(l)} = g(\Theta_{1.0}^{(l-1)} a_0^{(l-1)} + \Theta_{1.1}^{(l-1)} a_1^{(l-1)} + \ldots + \Theta_{1.k(l-1)}^{(l-1)} a_{k(l-1)}^{(l-1)})$$
$$a_2^{(l)} = g(\Theta_{2.0}^{(l-1)} a_0^{(l-1)} + \Theta_{2.1}^{(l-1)} a_1^{(l-1)} + \ldots + \Theta_{2.k(l-1)}^{(l-1)} a_{k(l-1)}^{(l-1)})$$
$$\ldots$$
$$a_{k(l)}^{(l)} = g(\Theta_{1.0}^{(l-1)} a_0^{(l-1)} + \Theta_{1.1}^{(l-1)} a_1^{(l-1)} + \ldots + \Theta_{1.k(l-1)}^{(l-1)} a_{k(l-1)}^{(l-1)})$$

So,

$$h_\Theta(x) = g(\Theta_{1.0}^{(L-1)}a_0^{(L-1)} + \Theta_{1.1}^{(L-1)} a_1^{(L-1)} + \ldots + \Theta_{1.k(L-1)}^{(L-1)} a_{k(L-1)}^{(L-1)}) \quad (1)$$

where,

L = total number of layers
l = 2 to L-1
$a_0^{(l)} = 1$ (Bias Unit)
$x_0 = 1$ (Bias Unit)
k(l) = number of units in l^{th} layer
$x_1, x_2, \ldots, x_n$ = input units
$a_i^{(j)}$ = 'Activation' of layer i in layer j
$\Theta^{(j)}$ = matrix of weights controlling function mapping from layer j to layer j+1
$\Theta_{j.i}^{(l)}$ = weight of the edge from i^{th} unit in layer l to j^{th} unit in layer (l+1)

3.3.3 Decoys

Traps are placed within the file system. The decoy files are downloaded by the legitimate user and placed in highly-conspicuous locations that are not likely to cause any interference with the normal user activities on the system. A masquerader, who is not familiar with the file system and its contents, is likely to access these decoy files, if he or she is in search for sensitive information, such as the bait information embedded in these decoy files. Therefore, monitoring access to the decoy files should signal masquerade activity on the system. The advantages of placing decoys in a file system are:

1. Detection of masquerade activity
2. confusion of the attacker and the additional costs incurred to distinguish real from bogus information, and
3. Deterrence effect which, although hard to measure, plays a significant role in preventing masquerade activity by risk-averse attackers.

Once a user registers with the system by providing a real email address and user chosen password, the user is ready to get started. Fog provides a number of tabs to select different actions. The "create decoy tab" is first used to request the generation of Adobe PDF or MS Word Doc files, etc each with an enabled "beacon" that executes whenever the document is opened. Once the user has generated a document, they may download it to their machine. The first time the document is downloaded, we suggest opening the document for review. The user will likely see a security pop-up message warning the user that the document is attempting to make a network connection with a server. The user should click on the box to *remember this action*, and then click on *allow* so that subsequently opening the document will not generate a pop-up message (which can be used by an inside attacker to detect the decoy). The document should then be stored in a conspicuous location such as the Desktop, or my documents folder.

In the future, any time the decoy is opened, an email will be sent to the registered email address warning the legitimate user that someone has opened a decoy document in their machine.

3.3.4 Information Gathered about Legitimate User

No personally identifiable information is gathered about the user. Users must register their email address with the site, an email address where they wish to receive alerts. The user's IP address is also noted by fog when a document is created. The document is associated with the email address and IP address and is used to inform the user whenever the decoys are opened. The IP address reported in the alert is the IP address of the host that opened the decoy document.

4. ALGORITHM

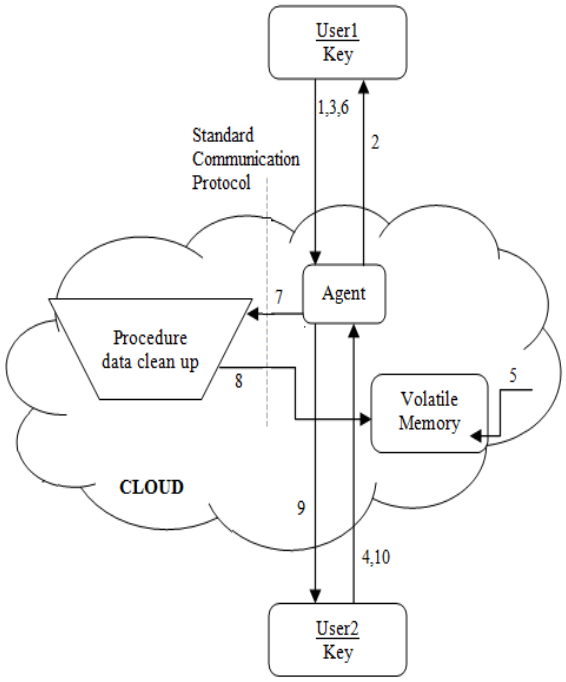


Fig.2. Selective Encryption Algorithm with Data Clearing

Figure 2 represents what happens when an insider tries to attack the system in the name of planned/unplanned maintenance and access the volatile memory where the customer has decrypted some data and is performing operations on it.

Steps followed in Figure 2:

1- User 1 contacts the agent requesting to access the cloud.
2 - Agent sends an acknowledgement to User 1.
3- User 1 authenticates itself with the agent and informs the agent about accessing encrypted data.
4 - User 2 requests access to a particular physical device for a planned/unplanned maintenance. Agent blocks this request.
5 – Data loaded to the volatile memory and decryption done.
6 – User 1 logs off/indicates completion task
7 – Agent triggers Procedure Data Clean Up.
8 – The volatile memory is cleaned.
9 – Agent sends acknowledgement to User 2.
10 – User 2 authenticates itself with the Agent and performs his operations.

5. IMPLEMENTATION DETAILS

The Insider threat attack was implemented initially by setting up a cloud environment using OpenNebula – an open source data center virtualization. A private cloud is necessary to demonstrate the attack, followed by, detection and prevention mechanism which includes the fog

computing conceptually. For testing we setup the private cloud in 4 VMs in the BTech Lab, NIT Trichy. For the user profiling algorithm, values for x1, x2, x3, etc. were chosen as described in section 3.

5.1 Software Specifications for the Cloud

1. Operating System – Ubuntu 12.04LTS
2. Packages - nfs-kernel-server, KVM Hypervisor, QEMU, libvirt, Open Nebula, sqlite, ttylinux, virish , ttylinux.
3. Applications – VNC viewer, RDP connection software
4. Windows XP SP3 – the virtual machine created here runs Windows XP SP3.
5. GCC 4.7.2, Octave 3.6.4

5.2 Software Specifications for the Client

1. Operating System – Ubuntu 12.04LTS
2. Applications –RDP connection software

OpenNebula is an open-source cloud computing toolkit for managing heterogeneous distributed data center infrastructures. The OpenNebula toolkit manages a data center's virtual infrastructure to build private, public and hybrid IaaS (Infrastructure as a Service) clouds.

OpenNebula assumes that your physical infrastructure adopts a classical cluster-like architecture with a front-end, and a set of cluster nodes where Virtual Machines will be executed. There is at least one physical network joining all the cluster nodes with the front-end. [8]

We installed OpenNebula across 17 machines and learnt User Patterns to generate a model for the Neural Network for a week. Our interface had an option of storing data that the client wanted in his own root directory that is password protected. He could choose to store the data in both an encrypted and an unencrypted way. For data stored in an encrypted way, we give them a plethora of options such as AES(Advance Encryption Standard)[9] , RSA[10] etc for text data and Chaos Theory based methods[11] for image and video data.

We wrote an agent in Java that would receive incoming requests and store them in a Priority Queue and process them accordingly. Log files for learning were stored in ROOT/LOG folder.

For every unencrypted file, a decoy was generated. We simulated the insider threat attack for both the encrypted and unencrypted data. The system was robust and secure.

6. CONCLUSION

Cloud computing security is ripe with new opportunities or future research, including cloud-related insider threats. The nature of the insider will change due to cloud computing impacts, but the opportunities for attacks will broaden. Researchers should take note of these new opportunities and respond accordingly to prevent, detect, and respond to new cloud-related insider attacks.

7. REFERENCES

[1] Y. Chen, V. Paxson and R.H. Katz, "What's new about cloud computing security," University of California, Berkeley Report No. UCB/EECS-2010-5 January, 20(2010).

[2] D. Eric and Shaw, “The role of behavioral research and profiling in malicious cyber insider investigations,” Digital Investigation, vol. 3, no. 1, pp. 20 – 31, 2006.

[3] F. Rocha and M. Correia, “Lucy in the sky without diamonds: Stealing confidential data in the cloud,” in Proceedings of the First International Workshop on Dependability of Clouds, Data Centers and Virtual Computing Environments, Hong Kong, ser. DCDV ’11, June 2011.

[4] Shenk J “Demanding More from Log Management system. SANS Insitute Whitepaper” 2008

[5] B. Salem and S. J. Stolfo, Keromytis A.D. - Computer sci. Dept., Columbia Univ., New York, NY, USA. “Fog computing: Mitigating Insider Data Theft Attacks in Cloud”, Security and Privacy Workshops (SPW), 2012 IEEE Symposium on 24-25 May 2012.

[6] J. Pepitone, “Dropbox’s password nightmare highlights cloud risks,” June 2011

[7] “Sealed Cloud – A Novel Approach to Safeguard against Insider Attacks”, Hubert A. Jäger, Arnold Monitzer, Ralf O. G. Rieken, and Edmund Ernst Uniscon, universal identity control GmbH, Agnes Pockels-Bogen 1, 80992 Munich, Germany

[8] Stanford Machine Learning Course by Andrew NG (https://www.coursera.org/course/ml April 22nd 2013)

[9] OPENNEBULA 3.4.1 - Windows XP VM creation and live migration. – By ANIL KUMAR (cloud.b.lab@zoho.com)

[10] D. Kahn,The Codebreakers, The Story of Secret Writing.New York: Macmillan, 1967

[11] Matthews R. On the derivation of a chaotic encryption algorithm. Cryptologia 1989;13:29–42.

A Novel Bio-Inspired Load Balancing of Virtual Machines in Cloud Environment

ASHWIN T. S, SHRIDHAR G. DOMANAL AND G. RAM MOHANA REDDY
DEPARTMENT OF INFORMATION TECHNOLOGY
NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA
SURATHKAL, MANGALORE, INDIA
{ashwindixit9, shridhar.domanal, profgrmreddy}@gmail.com

***Abstract* — Load Balancing plays an important role in managing the software and the hardware components of cloud. In this present scenario the load balancing algorithm should be efficient in allocating the requested resource and also in the usage of the resources so that the over/underutilization of the resources will not occur in the cloud environment. In the present work, the allocation of all the available Virtual Machines is done in an efficient manner by Particle Swarm Optimization load balancing algorithm. Further, we have used cloudsim simulator to compare and analyze the performance of our algorithm. Simulation results demonstrate that the proposed algorithm distributes the load on all the available virtual machines uniformly i.e, without any under/over utilization and also the average response time is better compared to all existing scheduling algorithms.**

Keywords* — *Load Balancing, Resource Utilization, Cloud Computing, Virtual machine, CloudAnalyst, particle swarm optimization.

I. INTRODUCTION

Cloud Computing is used both in industry and academia to store and retrieve the necessary documents and files. With its infrastructure, platform and software as services, it provides on demand computing resources for various applications [1]. The Cloud computing deployment models are broadly divided into two groups: (1) public cloud (2) private cloud. In cloud computing we offer a set of resources (i.e. hardware and software resources) as a service to the user but not as a product. These services are broadly classified into three types: i. Platform as a Service (PaaS) ii. Software as a Service (SaaS), and iii. Infrastructure as a Service (IaaS). If these three services can be sold to customers on the internet then it is called public cloud, whereas the data centers with sophisticated network to provide services to limited number of end users are referred as Private cloud. Some of the popular public cloud providers are Amazon Elastic Cloud (EC2)[2], Google App Engine. The combination of private cloud and the public cloud constitutes the hybrid cloud.

A Soft Computing technique called Particle Swarm Optimization (PSO) is used as an optimization method for finding the Global best solution, which is obtained by improving the local best solution in each iteration with respect to a value computed by the fitness function [3].

In general PSO algorithm is a computational method which optimizes the given problem by iteratively improving the candidate solution to a given measure of quantity.

In this paper, we focus on resource allocation of virtual machines and user requests. The main issue is to design an efficient scheduling algorithm which addresses the constraints such as heterogeneity, reliability, high communication delay etc., with low response time. We present a modified PSO algorithm for assigning all incoming jobs uniformly among the available virtual machines in an efficient way. For existing scheduling algorithms such as Round Robin and Throttled algorithms we compute and compare the response time for serving the incoming jobs.

The rest of the paper is organized as follows: Section II contains the background and related work, Section III gives the proposed algorithm, Section IV contains details about experimental setup, Section V contains the results and analysis and Section VI gives the conclusion.

II. BACKGROUND AND RELATED WORK

In cloud computing environment there are many algorithms which are used for scheduling and load balancing. The main focus is to assign all incoming jobs among the available virtual machines with minimal response time. Load balancing is defined as a process of making effective resource utilization by reassigning the total load to the individual nodes of the collective system and thereby minimizing the response time of the job. There are many algorithms in use; few are discussed below [4].

The Stochastic Hill Climbing algorithm for balancing the load has been developed by Brototi Mondal et al. [5]. This algorithm is similar to the genetic algorithm in which a simple iterative procedure continuously moves in the direction of an increasing value called UpHill, on reaching the peak it stops where no neighbor has a higher value. A random function is used to choose the variant while computing the uphill moves and the probability of the selection can vary with the steepness of the uphill move. By making minor changes to the computed values, it maps them to a set of other values and starts with the next iteration with the best element of the set. This process is repeated until we either reach the solution or the stopping criteria. Even though the results are better than round robin and FCFS it is an incomplete approach to solve such optimization problems.

The Modified Throttled algorithm developed by Shridhar G. Domanal and G. Ram Mohana Reddy [6] deals with two major divisions, the index of the virtual machine (VM) and the state of the VMs. For every client request, a VM is selected

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

depending upon the state of the VM ID or -1 is returned to the data center. In their earlier work when the next request arrives the VM next to it in the index which is already assigned is selected depending upon its state. They further modified it by referring the request count on each VM .The least loaded VM is considered for the allocation and it also checks whether it is used in the previous assignment or not.

There are many standard algorithms such as round robin algorithm in which, upon arrival of the first request the data Centre randomly picks a VM from the group and assigns it in a circular order. Then the assigned VM is moved to the end of the list. The drawback here is that the incoming job should wait in the queue if any of the VMs are not free and the processing time for each individual request is not considered [7].

In Throttled Load Balancer or Throttled algorithm, the client first requests the load balancer to find a suitable Virtual Machine to perform the required operation [8].

In this paper we are comparing the simulated results obtained from our algorithm with the results from scheduling algorithms such as Round Robin and Modified Throttled algorithms. Our main focus is to improve the efficiency and distribute the load uniformly among the available VMs using Proposed PSO Load Balancing algorithm.

III. Proposed PSO Load Balancing Algorithm

The main focus of our proposed algorithm is to allocate the incoming requests to the virtual machine intelligently. The Basic methodology of the proposed algorithm is as shown in the below given Figure 1.

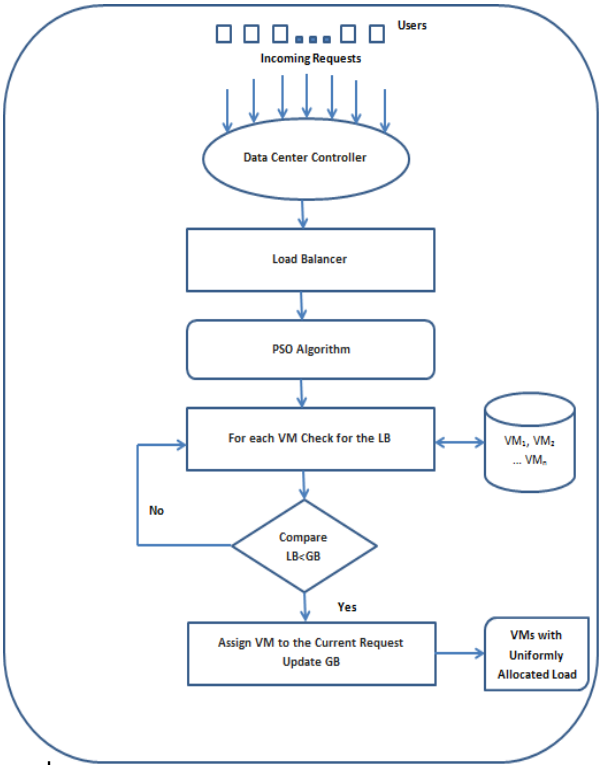


Figure 1: Flow of proposed PSO Algorithm

Initially all the virtual machines status i.e, availability, load already held by the VM is obtained and stored in the VM count and for each iteration, the VMCount is checked from the virtual machine and assigned to the local best. Then this local best is compared with the Global Best, if the local best is less than or equal to the global best, the corresponding virtual machine is assigned with the load and the data center proceeds to serve the next request in the same way. Meanwhile the global minimum i.e., the Global best is also updated. If the global best is less than the local best, the algorithm iterates until the given criteria is matched.

Algorithm: VM PSO Load Balancer

Input: Number of incoming jobs X1, X2… Xn.
Available Virtual Machines (VM) Y1, Y2… Yn.

Output: All incoming jobs X1, X2… Xn are allocated to each virtual machine in such a manner that the load on each virtual machine is distributed among Y1, Y2… Yn uniformly.

1: Initially all the VMs have 0 allocations.
Thus LB and GB are set to zero

2: VM PSO load balancer maintains the index / assign table of VMs which has the number of requests currently allocated to each VM.

3: Index table is parsed and least loaded VM is selected and correspondingly Local Best (LB) is compared with the VMCount of the VM.

Case: if LB ≤ GB

a. **if YES**
VM is chosen for assigning the incoming job request
Update GB

b. **if NO**
Repeat step 3 until case **a** is valid

4: VM PSO load balancer updates the data center by returning the VM id to the data center.

5: Request is assigned to the VM after updating the Global Best (GB).

6: Data center notifies the proposed PSO load balancer about the allocation.

7: The request held by each VM is updated by PSO load balancer.

IV. Experimental Set Up

The complete experiment is carried out using Cloud Analyst simulator. The cloud infrastructure is distributed; requests should be handled from different geographical locations. This simulator supports experiments in real time.

Brief overview of simulator: Cloud Analyst

Cloud Analyst simulator gives the real time scenario with six different geographical locations, i.e. no. of users from particular locations can be identified depending on the specific application, e.g. Facebook users from Asia, Africa etc. The simulator is very flexible and provides data centers, virtual machines, band width and many more resources for experimentation [9].

A snapshot of the Cloud Analyst architecture is shown in Figure 2.

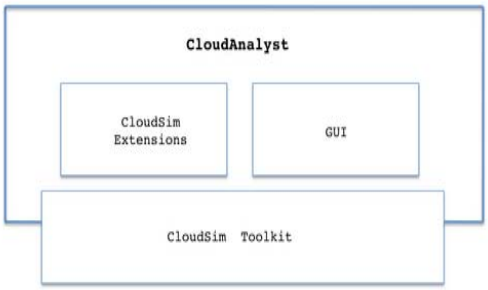


Figure 2: Cloud Analyst Architecture

For the experiment we have considered applications like Facebook, Twitter and Internet users. Six different continents of the world which signify six different geographical locations are considered [3]. A single time zone is considered for all user locations. For simplicity one hundredth of the total users from each continent is considered and it is assumed that only 5% of total users are online during peak hours and in off peak hours, users are one tenth of the peak hours as shown in table 1.

We have considered internet users of different continents for the month of June 2012[6]. The same data is experimented with three different scheduling algorithms and response time of each algorithm is considered for the result analyses.

TABLE 1: SIMULATED CONFIGURATION

USER BASE	REGION	SIMULTANEOUS ONLINE USERS DURING PEAK HOURS	SIMULTANEOUS ONLINE USERS DURING OFF HOURS
NORTH AMERICA	0	135000	13500
SOUTH AMERICA	1	125000	12500
EUROPE	2	255000	25500
ASIA	3	535000	53500
AFRICA	4	30000	3000
OCEANIA	5	10000	1000

Every Data Center has a capacity to host a number of virtual machines which are needed for particular application. Machines have 100 GB of storage space, 4 GB of RAM, each machine has 4 CPUs and a power of 10k MIPS [10].

V. RESULTS AND ANALYSIS

The obtained results are analyzed and then compared with scheduling algorithms like Modified throttled load balancing and Round Robin. This comparison is done based on the efficient utilization of the VMs by avoiding the under/overloading conditions and the average response time of each VMs.

A. Load balancing of VMs

In an Active Load Balancer algorithm, the least loaded VM is assigned, depending on its current load. Hence few VMs are overloaded with many requests and remaining VMs are underutilized with few requests to handle. This results in the imbalance of load in VMs. But with our proposed algorithm all the virtual machines are utilized properly and completely. By using our algorithm we can observe that there is no underutilization of the resources in cloud. Initial configuration used 5 VMs for testing and subsequently with 25 and 50 VMs. In all the cases, we can observe efficient utilization of all the available VMs. Table 2 is a comparison of allocation of load on each VM using two different algorithms. One is the Modified throttled algorithm and the other is the proposed PSO Load Balancing algorithm.

B. Response time of incoming requests

The data set from Table 1 is used as the set of incoming requests and we observe that the average response time of the proposed PSO algorithm is less than other scheduling algorithms. The Table 3 gives the average response time of different scheduling algorithms.

TABLE 2: 5 VMS USAGE

SL.NO	MODIFIED THROTTLED	PROPOSED PSO LOAD BALANCER
VM0	255	254
VM1	255	254
VM2	253	254
VM3	253	253
VM4	252	253

Similarly we observe that the allocation of VMs by Proposed PSO load balancing algorithm is better than the existing Round Robin and Throttled algorithms. Further it is also observed that the average response time of the proposed algorithm is better than other scheduling algorithms (Table 3).

TABLE 3: ANALYSIS OF AVERAGE RESPONSE TIME

ALGORITHM	AVERAGE RESPONSE TIME
Round Robin	364.85 ms
Modified Throttled	363.52 ms
Proposed PSO	360.11 ms

This comparison of average response time of the three different scheduling algorithms are plotted in a graph which is as shown in the Figure 3

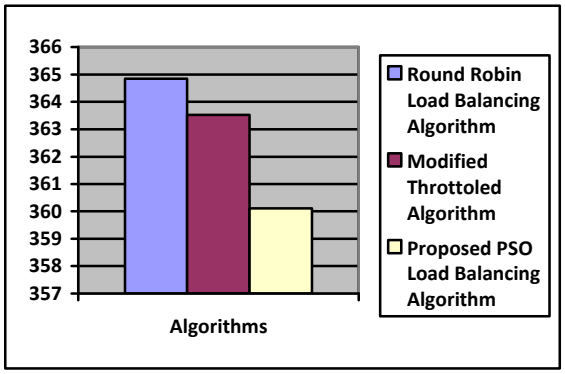


Figure 3: Analysis chart of average response time

From the results obtained we can observe that the proposed PSO Load Balancing Algorithm is not only efficient in terms of the allocation of load on the available VMs uniformly but also efficient in terms of the average response time.

VI. Conclusion

The proposed PSO Load balancing Algorithm is an efficient algorithm which performs better than the existing load balancing algorithms. It manages the load at the server and intelligently assigns it to all the available VMs by considering its status. The efficient utilization of the resources is the main motto of the proposed PSO load balancing algorithm. With simulation we have proved that the proposed algorithm will load without any under/overutilization of the available VMs. As we can observe from the table 3 that not only the proposed algorithm shows its efficiency in under/overutilization but it is also efficient in terms of the average response time.

In the future, the given proposed algorithm can be optimized by taking both static and dynamic loads simultaneously into account and also by taking scenarios in which the incoming requests are rapidly varying.

Acknowledgment

We like to acknowledge Subhayan Mukherjee, IT Department NITK for his precious and timely suggestions on this paper.

References

[1] Xu Wang ; Beizhan Wang ; Jing Huang;" Cloud computing and its key techniques" 2011 IEEE International Conference.

[2] Wenhua Zeng ; Jianfeng Zhao ; Min Liu :"Several Public Commercial Clouds and Open Source Cloud Computing Software" Computer Science & Education (ICCSE), 2012 7th International Conference.

[3] Eberhart, R. C., and Kermedy, J. (1995). "New Optimizer Using Particles Swarm Theory", Proc. Sixth International Symposium on Micro Machine and Hmm Science (Nagoya, Japan), IEEE Service Center, Pkcataway, NJ 39-43.

[4] Domanal, S.G. ; Reddy, G.R.M." Optimal load balancing in cloud computing by efficient utilization of virtual machines" Communication Systems and Networks (COMSNETS), 2014 Sixth International Conference

[5] Brototi Mondal, Kousik Dasgupta and Paramartha Dutta, "Load balancing in cloud computing using stochastic hill climbing-a soft computing approach" in Procedia Technology 4 (2012) 783 – 789, ELSEVIER C3IT-2012Wickremasinghe, B. ;

[6] Shridhar G. Domanal and G. Ram Mohana Reddy," Load Balancing in Cloud Computing Using Modified Throttled Algorithm" IEEE, International conference. CCEM 2013. In press

[7] Mr.Manan D. Shah, "Allocation Of Virtual Machines In Cloud Computing Using Load Balancing Algorithm" in International Journal of Computer Science and Information Technology & Security (IJCSITS), ISSN: 2249-9555 Vol. 3, No.1, and February 2013.

[8] Ms.Nitika, Ms.Shaveta, Mr. Gaurav Raj, "Comparative Analysis of Load Balancing Algorithms in Cloud Computing" in International Journal of Advanced Research in Computer Engineering & Technology Volume 1, Issue 3, May 2012.

[9] Calheiros, R.N. ; Buyya, R "CloudAnalyst: A CloudSim-Based Visual Modeller for Analysing Cloud Computing Environments and Applications" in 2010 24th IEEE International Conference on Advanced Information Networking and Applications

[10] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, C´esar A. F. De Rose and Rajkumar Buyya "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms" 24 August 2010 In Wiley Online Library.

Achieving Energy Efficiency by Optimal Resource Utilisation in Cloud Environment

Devwrat More [a]
moredm09.comp@coep.ac.in

Sanket Mehta [a]
mehtass10.comp@coep.ac.in

Pooja Pathak [a]
pathakpk10.comp@coep.ac.in

Lokesh Walase [a]
walasels09.comp@coep.ac.in

Jibi Abraham [a]
ja.comp@coep.ac.in

[a] - Department of Computer Engineering and IT, College of Engineering Pune, India.

Abstract— **Emergence of cloud computing has provided an efficient platform for distributed utility computing. But, the huge amount of energy consumed in cloud environment has raised many environmental concerns. To solve the issue of energy consumption in cloud environment, the design of energy efficient mechanism must start to play a major role. The paper addresses this issue by proposing *Energy Manager* which governs activities in cloud to attain optimal energy utilisation. The Energy Manager takes into account resource utilisation, CPU frequency, supply voltage and effect of all these on response time. In a novel way, it uses soft scaling as a precursor to Dynamic Voltage and Frequency Scaling (DVFS) to enhance the effectiveness of DVFS in virtualized environment. It implements DVFS, Virtual Machine (VM) migration and consolidation, soft scaling, and power state switching techniques to achieve the objective.**

Keywords- **Cloud Computing, Green Cloud, Energy Manager, Energy Efficiency, Resource Utilisation**

I. INTRODUCTION

Enterprise and research programs create growing demand for computing resources. To satisfy this need, high performance computing with quality resources was needed to be developed. Cloud computing [1] is the solution developed for addressing the above need. It is a model for enabling ubiquitous, convenient and on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

The exponentially growing computational demand in industry raised the need to utilise hardware efficiently. Even though the requirement of hardware increased, buying hardware individually and maintaining it was not affordable for customers. Data centres having a cluster of high end servers with advanced hardware, availed the computing resources for this computational requirement. As typical resources (e.g. memory, storage, etc.) in data center are highly configured than required to satisfy the needs of a single customer, appropriate sharing of resources by allowing more than one customer to use a single resource became necessary.

Virtualisation emerged as a solution to this problem and at the same time assured security. Now it has become the fundamental part of cloud computing. Virtualization, in computing, is a term that refers to the various techniques, methods or approaches of creating a virtual (rather than actual) version of something, such as a virtual hardware platform, operating system (OS), storage device, or network resources.

Data centres which are building blocks of cloud computing services are used on extensive scale in modern IT industry. Data collected from more than 5000 production servers over a six-month period showed that servers operate only at 10-50% of their full capacity most of the time, leading to expenses on over-provisioning, and thus extra

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

Total Cost of Acquisition [4]. The challenging problem of the narrow dynamic power range of servers i.e. even completely idle servers still consume about 70% of their peak power [5], along with the above statistics motivated a thought for identifying the efficient ways of energy utilisation in cloud. A number of practices can be applied to achieve energy-efficiency, such as improvement of applications' algorithms, energy efficient hardware, Dynamic Voltage and Frequency Scaling (DVFS) [6], terminal servers and thin clients, and virtualization of computer resources [7]. Currently, most of the techniques of energy efficiency like DVFS are designed considering energy management of a single host. Thus, as it is implementation of such techniques becomes inefficient in cloud due to failure in consideration of global scenario.

Lowering energy consumption by such methodologies can hamper the performance of cloud resulting in increased response time. Increased response time may result in violation of Service Level Agreement (SLA) between cloud vendor and client. Thus, managing energy efficiently with consideration of SLA requirements is a challenging problem. These approaches along with techniques like VM migration, VM consolidation, process migration, hard and soft scaling can be combined to develop the proposed energy efficient cloud.

This paper proposes an approach to reduce energy consumption in cloud environment using optimal resource utilisation. Section 2 gives an overview of the work related to our proposed system. Section 3 explains the overall system design and each module of the system in detail. Section 4 analyses the results observed during the implementation. Conclusion and future work of the paper is mentioned in Section 5.

II. RELATED WORK

There are many techniques in the literature which achieve energy efficiency. Techniques like DVFS and power state switching are implemented considering the state of a single node whereas VM migration and consolidation are implemented considering the state of overall cloud environment.

VM migration refers to the process of moving a virtual machine between different physical machines by transferring memory, storage and network connectivity of the VM from original host machine to the destination machine. Set of algorithms have been proposed for VM migration and consolidation to achieve energy efficiency in cloud [9]. This paper also shows how various heuristic techniques like "single threshold" and "highest potential growth" impacts the energy consumption of cloud environment.

Power state switching techniques include switching a node into an appropriate power state. There are two power management schemes. In static power management, the system defines several sleep modes with various levels of energy saving and delay overhead. In dynamic power management, system automatically detects the idle period and disables the clocks on portions of the CPU [8]. These techniques are node based techniques which can be implemented in cloud environment with modifications.

DVFS based approaches monitor the CPU usage and accordingly change the CPU frequency. DVFS is implemented by the subsystem of the Linux kernel called as *cpufreq*. The *cpufreq* infrastructure allows for different frequency-changing policy governors. *Ondemand and conservative* governors change the CPU frequency depending on CPU utilisation [2]. One of the major shortcomings of using these governors in virtualized environments like cloud is, even if a node is underutilised, a VM running on it may have high load depending on the amount of resource allocated to it. In this scenario, these governors reduce the frequency of the node which further hampers the performance of VM. Another reason why these governors are not very effective in cloud environment is that for changing frequency of the host, overall load in the cloud is not considered.

Soft scaling is a technique which entails setting the time quantum of every VM running on the same CPU core in proportion to the workload. In literature, this technique is mainly used to achieve maximum resource utilisation which in turn helps managing response time of a VM in a better way [3].

III. PROPOSED SYSTEM

Proposed solution attempts to achieve optimum energy utilisation in cloud environment. Our approach tries to overcome flaws mentioned in the related work by considering utilisation at three levels: VM level, host level and cloud level. The proposed solution uses VM migration and power state switching techniques. In combination with this, each host is set to appropriate frequency and voltage using DVFS technique. The use of Soft scaling technique enhances the energy optimization effect of DVFS.

By combining all above approaches, the proposed solution - *Energy Manager*, ensures to achieve the goal of minimum energy consumption with consideration of SLA.

System Design:

After detailed study and analysis of related work in this domain, it was found that a considerable amount of work

has been done in developing various techniques to reduce energy consumption in cloud environment. However each of them has been implemented independently and there has not been much effort that intelligently combines these techniques. The *Energy Manager* proposed here not only takes intelligent decision about using these techniques, but also uses soft scaling in a novel way to enhance energy savings using DVFS.

The *Energy Manager* has four modules. The brain of the *Energy Manager* i.e. "Decision Maker" (DM) module not only decides most appropriate combination of the techniques to be used but also implements soft scaling as energy reduction technique. To make an intelligent decision, the DM module analyses the current situation in the cloud. DM needs certain parameters for this purpose. These parameters are gathered periodically by another module, called as "Utilisation Supervisor" (US). A module "Virtual Machine Allocator" (VMA) is needed to attend the request for new virtual machine in the cloud. "Implementer" (IM), the fourth module implements intelligent decisions made by DM and VMA.

Fig.1 shows the different modules of the proposed system and the way they interact with each other.

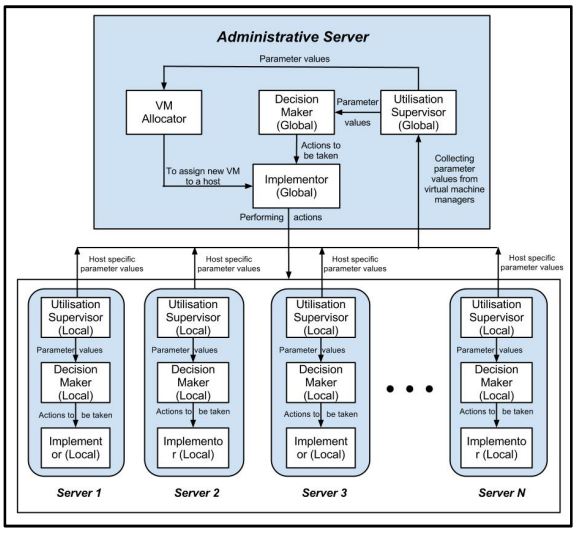


Figure 1: System design diagram

A. Utilisation Supervisor :

Utilisation Supervisor (US) is a monitory module which periodically collects parameters from various hosts. Based on the collected parameters, US analyses current state of physical hosts and virtual machines. Then it takes a decision of invoking DM if there is a scope for reducing energy consumption through resource management. For example, if US finds that more than required number of physical hosts are running, then it would invoke DM.

US resides at two levels in cloud: Local US and Global US. Local US resides on every host to collect host level parameters. The parameters include frequency and utilisation of each CPU core, number of VMs running on each CPU core, the utilisation of each VM and the soft scaling attributes of VM (i.e. VM cap and CPU affinity [10]). Global US running on the administrative server collectively analyses data from all local US and in turn invokes global DM if necessary.

Utilisation of VM is dependent on frequency of corresponding core, utilisations of VMs running on different cores cannot be compared directly. Example: Consider two VMs running on different cores, showing same CPU utilisation. But if both cores are running at different frequencies actual VM utilisations can't be considered as equal. To compensate the effect of frequency on VM utilisation, VM utilisation value is multiplied with weight assigned to frequency at which the core is running. Thus **W**eighted **U**tilisation of **V**M is calculated using Equation (1). Equation (1) makes an assumption that when the frequency is doubled, load gets reduced to half.

$$\boldsymbol{WU}_{\mathrm{Vi}} = \frac{\boldsymbol{f}_{\mathrm{curr}}}{\sum_{\boldsymbol{i}} \boldsymbol{f}_{\boldsymbol{i}}} * \boldsymbol{UV}_{\mathrm{i}} \qquad (1)$$

Where,
WU_{Vi} = Weighted Utilisation of i^{th} VM
f_{cur} = frequency of corresponding core
$\sum f_i$ = Summation of all available frequencies

Every host has an upper limit on amount of load it can carry. If this limit is crossed it in turn increases response time hampering the SLA. While calculating this upper limit on host (C), instead of considering maximum core utilisation as 100%, a buffer in the form of delta (δ) is considered, which accommodates for instantaneous surges in load. This avoids unnecessary SLA violations. As per equation (1), upper limit of load is computed in terms of weighted utilisation as well, which considers the highest frequency weight as a factor for upper limit.

$$\boldsymbol{C} = [(\mathbf{100} - \boldsymbol{\delta}) * \boldsymbol{Wf}_{\mathbf{h}}] * \boldsymbol{N}_{\mathrm{cores}} \qquad (2)$$

Where,
C = Upper limit on host
δ = buffer to accommodate surges
Wf_h = Weight of maximum frequency
N_{cores} = No of cores available on host

Global US calculates the required number of hosts (H_{req}) for the current load using (3).

$$\mathbf{H_{req}} = \sum_{i=1}^{N_H} \sum_{j=1}^{N_{Vi}} \mathbf{WU_{Vij}} \Big/ \mathbf{C} \qquad (3)$$

Where,
N_H = Number of hosts
N_{Vi} = Number of VMs on i^{th} host

Global US compares if number of hosts actually running in the cloud is same as H_{req}. If not, it invokes DM with flag indicating need of power-state switching mechanism which defines three power states: On, Sleep, Off.
If required number of hosts are running in the cloud as per above calculations, US checks whether the load is evenly distributed amongst the hosts, using the calculated average load (L_{avg}) (see equation (4)). US then assigns Migration or Safe band to each host, depending on load on that host. Migration band and safe band are defined using Δ (see equation (5)) as shown in the figure 2. If any of the hosts running in the cloud, fall in the migration band (see figure 2), US invokes Global DM with flag indicating need of load balancing.

$$\boldsymbol{L_{avg}} = \sum_{i=1}^{N_H} \sum_{j=1}^{N_{Vi}} \boldsymbol{WU_{Vij}} \Big/ \boldsymbol{H_{running}} \qquad (4)$$

Where,
N_H = Number of hosts
N_{Vi} = Number of VMs on i^{th} host

$$\Delta = \mathbf{50\%\ of\ WUV_{avg}} \qquad (5)$$

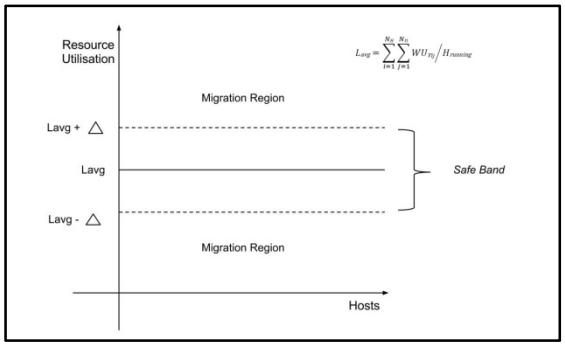


Figure 2: Band based load distribution

If all hosts lie in safe band, Global US exits without invoking DM indicating current cloud environment is ideal for energy conservation.

B. Decision Maker :

This is an intelligent module of Energy Manager which is responsible for optimal energy consumption. Just like Utilisation supervisor it has two parts: Local and Global DM. Global DM is invoked by Global Utilisation supervisor depending on situation in cloud.

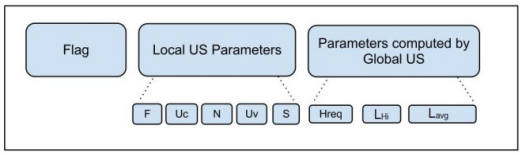


Figure 3: US-DM Communication Protocol

Figure 3 shows the communication protocol between Global US and DM. It consists of following three fields:

- **Flag:** The flag raised indicates the need for either switching power state or fair distribution of load.
- **Local US Parameters:** It consists of frequency (F), Utilisation per core (Uc), Number of VMs (N), Utilisation per VM (Uv), Soft scaling attributes (S).
- **Parameters computed by Global US:** It consists of required number of hosts (H_{req}), Load on each host (L_{Hi}) and average load (L_{avg}).

According to this communication protocol, information is passed from US to DM which aids DM in decision making.
When DM is invoked for switching the power states of hosts, it first checks if H_{req} is less than or greater than $H_{running}$ (Number of running hosts),

When H_{req} is greater than $H_{running}$, DM randomly selects $(H_{req} - H_{running})$ number of hosts from pool of hosts in lower power states to turn on. If H_{req} is less than $H_{running}$, the DM identifies $(H_{running} - H_{req})$ number of least loaded hosts for lowering the power state. The load on these machines is then shifted to other machines. For this, VMs are allocated to hosts using many-to-one mapping. Thus, each VM is allocated a host.

After handling power state switching to achieve H_{req} number of running hosts in cloud, load distribution logic of DM is called, which is directly called if the flag in US-DM communication reflects DM is invoked because of unfair load distribution. This mechanism considers all the hosts present in the migration band as migration subjected hosts. The mechanism arranges the VMs amongst migration subjected hosts, in such a way that maximum possible number of migration subjected hosts would be shifted from migration band to safe band. While performing VM migration, it is ensured that the VM resource requirements are satisfied by the resource availability on the allocated host. If not, then another target is chosen according to VM configuration requirement. Thus Global DM ensures even load distribution on required number of hosts.

Local DM is a periodically invoked module. It handles host-level energy efficiency techniques like scaling frequency of CPU core dynamically depending on load on it. It also dynamically manipulates soft scaling attributes of VM i.e. time-quantum assigned to VM, depending on VM utilisation. In case of the Xen hypervisor's *Credit Scheduler*, *VM Cap* is the attribute which allows restriction on time quantum used by VMs. Thus, to allocate CPU core optimally, cap of each VM is calculated depending on the VM utilisation using equation (6).

$$C_i = \left(U_{Vi} \Big/ \sum_{j=1}^{k} U_{Vj} \right) * 100 \qquad (6)$$

Where,

C_i = *Cap of VM*

U_{Vi} = *Utilisation of* i^{th} *VM on the CPU core*

k = *Number of VMs on the CPU core*

Thus, this technique - soft scaling, is used as a precursor for more effective hard scaling. For example, consider a scenario where two VMs are running on the same core with one VM overloaded and another one idle. As the CPU core is equally distributed among these two VMs, the effective CPU utilisation is less. In such scenario, the core frequency should be reduced using hard scaling. But this will in turn increase the response time of overloaded VM, thus violating the response time constraints in SLA. In such case, using soft scaling, more CPU time quantum can be allocated to the overloaded VM, satisfying its SLA requirements. Now, as the VM requirements are satisfied, hard scaling can be performed to reduce energy consumption.

Local DM also performs the load distribution at host level. By considering the load on each VM on the host, the Local DM sets the CPU affinity of the VCPUs of VMs in such a way that each core has approximately the same load. This load distribution problem is solved as balanced partitioning problem at core level. Thus achieving, even load distribution on all cores.

Credit Scheduler of Xen performs automatic load balancing on all cores, which becomes hindrance in applying DVFS. For example, consider a case of 4 VMs are running on a host (with 4 cores), first VM utilising 70% CPU and the rest hardly using any. Now, the default credit scheduler will keep moving the first VM on each core of the host, and hence, apparently, there is no scope for frequency reduction. But, the Local DM of *Energy Manger* pins the first VM to one core and now, uses DVFS to reduce the frequency of other cores.

The decisions of both Local and Global DM are passed to IM for implementation.

C. Virtual Machine Allocator :

VMA is invoked when there is a request for a new VM from client. VMA then decides the target host which can accommodate the newly requested VM. An average resource utilisation is assumed for the new VM. With this assumption and the given configuration of the VM, VMA recalculates H_{req}. If H_{req} is less than $H_{running}$, $(H_{running} - H_{req})^{th}$ least loaded host is selected as the target host. This avoids allocating a power switch prone host to the new VM. If H_{req} is greater than $H_{running}$, a host from low power state pool is randomly selected as the

target host. The VMA sends information of new VM and target host to the IM. If is H_{req} same as $H_{running}$, the least loaded host is selected as the target host.

D. Implementer:

IM directly connects all servers in the cloud. As the actions are to be implemented at both host and cloud level, IM also functions at two levels: Local IM which implements actions as conveyed by local DM and Global IM which is responsible for carrying out the actions of VMA and the Global DM.

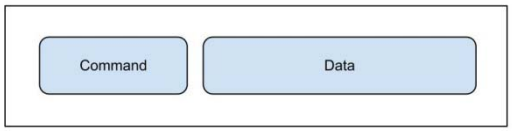


Figure 4: IM Communication Protocol

Figure 4 depicts the general protocol used by DM and VMA while communicating with IM. The modules use a shared file wherein communication parameters are written in the format shown above.

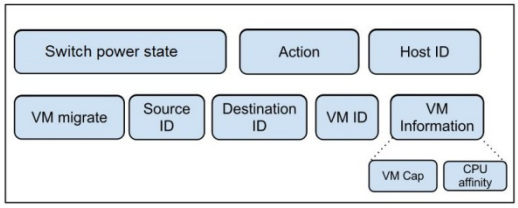


Figure 5: DM-IM Communication Protocol

Figure 5 shows the specific fields when DM communicates with IM. The command given by DM can either be "Switch power state" or "VM migrate".

If the command is "Switch power state", the fields are:

- **Action:** Turn on, Sleep or Turn off.
- **Host ID:** ID of the host whose power state is to be changed.

When the command is "VM migrate", the fields are:

- **Source ID:** ID of parent host on which VM currently resides.
- **Destination ID:** ID of the target host, where the VM has to be migrated.
- **VM ID:** ID of the VM which is to be migrated.
- **VM information:** VM Cap and CPU affinity.

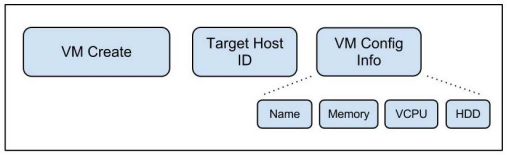


Figure 6: VMA-IM Communication Protocol

Figure 6 shows the specific fields using which VMA communicates with the IM. The command issued by the VMA is "VM create".

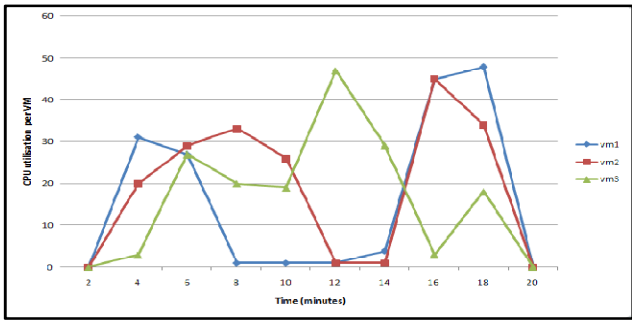
- **Target Host ID:** ID of the host where the VM is to be migrated.
- **VM Config Info:** name of the VM, memory, storage and the number of virtual CPUs allocated to the VM.

All the above decisions are taken with consideration of SLA. Thus, *Energy Manager* not only reduces energy consumption but also tries to maintain SLA requirements.

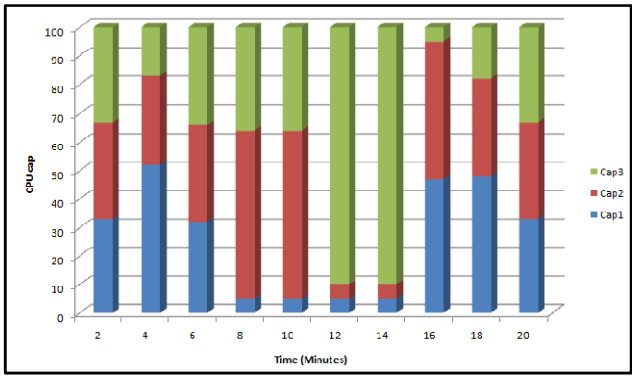
IV. EVALUATION AND ANALYSIS

To verify the effect of proposed "Energy Manager", various experiments were conducted which were categorised in 2 parts: experiments at host level and experiments at cloud level.

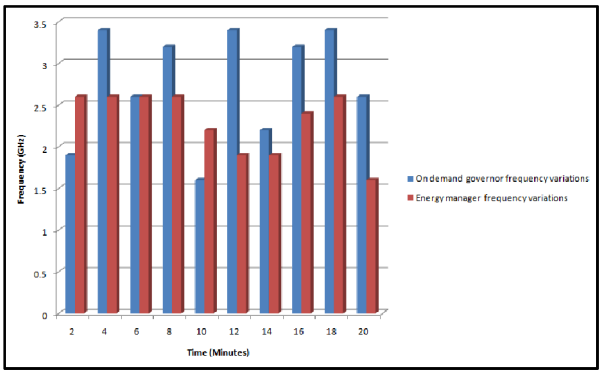
To evaluate the effect of local Energy Manager, host level experiments were carried out on Intel(R) CoreTM i7-2600 CPU @ 3.40GHz * 4, 4 GB memory and 500 GB storage with Ubuntu-12.04 as dom0 OS using Xen hypervisor. 5 VMs were created with Ubuntu-12.04 as OS, 512 MB memory and 2 GB of storage. Variable amount of load was created on each VM. The exact load scenario was executed twice, with and without the local Energy Manager running on dom0. This setup was observed for 20 minutes, where in following parameters were measured: CPU utilisation per VM, cap assigned to each VM, frequency variations of the CPU, temperature of the CPU (using P-Sensor software), and Power and Energy (using Energy Meter). Following graphs depict the obtained results. Graph 6 shows that roughly 20% of the energy was saved during the experiment.



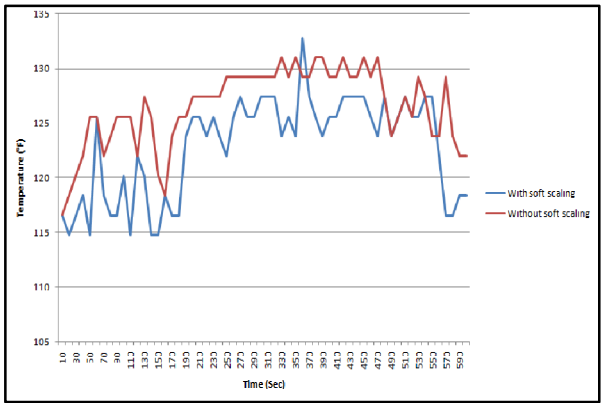
Graph 1: CPU Utilisation per VM



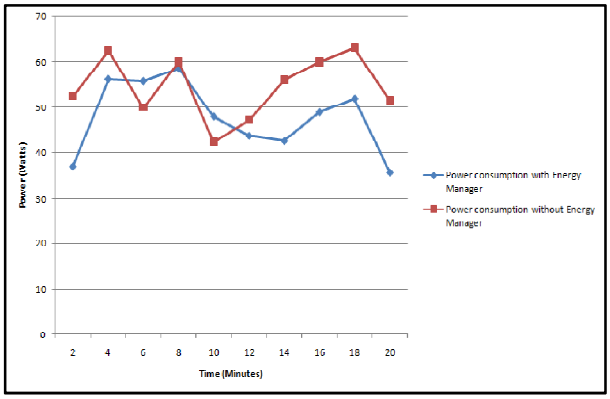
Graph 2: CPU Cap per VM



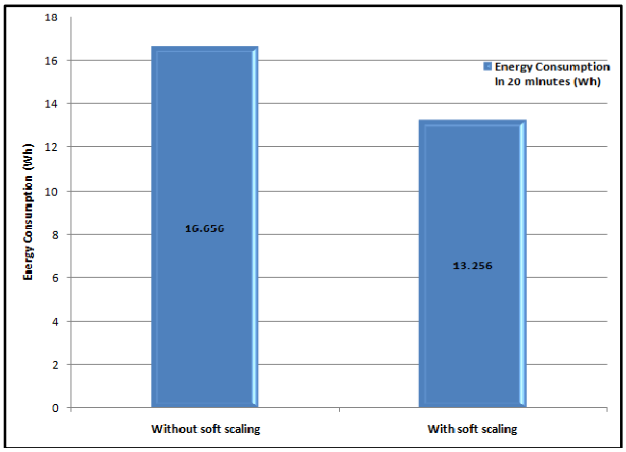
Graph 3: Frequency Variations



Graph 4: Effect of local Energy Manger on temperature



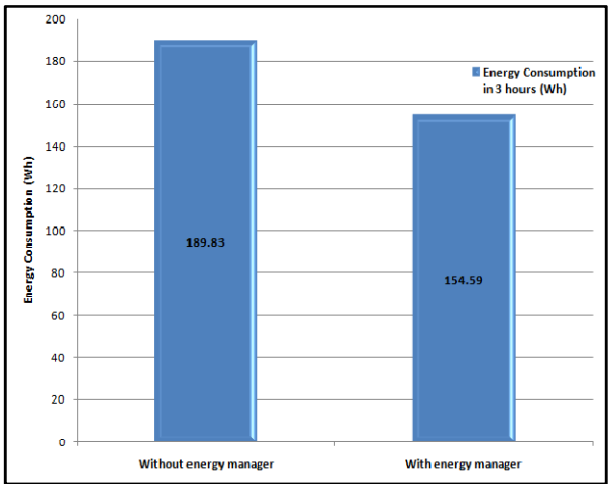
Graph 5: Effect of Local Energy Manager on Power



Graph 6: Effect of Local Energy Manager on Energy Consumption

To analyse the effect of complete Energy Manger in cloud environment, an experiment was performed at cloud level. A test bed consisting of 5 hosts (of the same

configuration) with each host having variable number of VMs (3-7) was created. One of the host was randomly chosen as the administrative host which executed the global module of Energy Manager. Local Energy Manager module was instantiated on each of the host. As with the previous experiment, random load was generated on each of the VMs and the environment was observed for 3 hours. Graph 7 shows the results indicating that roughly 19% energy was saved.



Graph 7: Effect of Energy Manager on Energy Consumption

V. Conclusion and Future Work

In this paper, we have presented the design, implementation and evaluation of Energy Manager. This approach used techniques like: VM migration, DVFS, Soft scaling and Power State Switching. Energy Manager by optimally allocating resources and switching off the unrequired resources achieves reduction in energy consumption. DVFS locally manages the CPU frequency and voltage by reducing energy consumption of each individual host. Load distribution at host level ensures total utilisation of available resources while load distribution at core level tries to attain minimum possible frequency and voltage across all cores. Thus, this not only enhances performance of Energy Manager in terms of energy conservation, but the fair distribution of load across data center also prevents formation of hot spots. Soft scaling makes sure that each VM is allocated appropriate amounts of resources, acting as and helps DVFS to achieve greater efficiency. Also, the additional *buffer* is considered at each step of algorithm design to respect the SLA restrictions. Thus, Energy Manger reduces energy consumption of a cloud environment with considerations for SLA requirements.

Currently, the Energy Manager tries to achieve optimal utilisation of CPU only. The same algorithms presented in current design can be extended for other resources like memory, I/O, network etc. There is always a scope to optimise available algorithms in terms of SLA violations. Current system uses VM migration for load balancing. Implementation of Process Migration in addition to that can lead to enhanced energy efficiency. Currently, Decision Maker uses instantaneous values of various attributes. It can be improved to make better decisions by analysing the behaviour of various components in cloud by using techniques of Machine Learning.

VI. References

[1] Mell, Peter, and Timothy Grance. "The NIST definition of cloud computing (draft)." *NIST special publication* 800 (2011): 145.

[2] Pallipadi, Venkatesh, and Alexey Starikovskiy. "The ondemand governor." *Proceedings of the Linux Symposium*. Vol. 2. sn, 2006.

[3] Nathuji, Ripal, and Karsten Schwan. "VirtualPower: coordinated power management in virtualized enterprise systems." *ACM SIGOPS Operating Systems Review*. Vol. 41. No. 6. ACM, 2007.

[4] L. A. Barroso and U. Holzle. The case for energy-proportional computing. Computer, pages 33–37, 2007.

[5] X. Fan et al. Power provisioning for a warehouse-sized computer. In Proc. of the 34th Annual Intl. Symp. On Computer Architecture, pages 13–23, 2007.

[6] G. Semeraro, G. Magklis, R. Balasubramonian, D. H. Al- bonesi, S. Dwarkadas, and M. L. Scott, "Energy-efficient processor design using multiple clock domains with dynamic voltage and frequency scaling," in Proceedings of the 8th International Symposium on High-Performance Computer Architecture, 2002, pp. 29–42.

[7] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," in Proceedings of the 19th ACM symposium on Operating systems principles, 2003, p. 177.

[8] Srivastava, Mani B., Anantha P. Chandrakasan, and Robert W. Brodersen. "Predictive system shutdown and other architectural techniques for energy efficient programmable computation." *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 4.1 (1996): 42-55.

[9] Beloglazov, Anton, et al. "A taxonomy and survey of energy-efficient data centers and cloud computing systems." *Advances in Computers* 82.2 (2011): 47-111.

[10] Li, Zhi, et al. "Affinity-aware dynamic pinning scheduling for virtual machines." *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*. IEEE, 2010.

An Enhanced Cloud Computing Model for Patient Record Management in South Africa

Richard Millham
Durban University of Technology
Richardm1@dut.ac.za

Abstract - **Cloud computing provides low initial cost for clients, scalability, and the ability to optimally use and share computing resources which makes cloud computing particularly attractive for emerging countries. In this paper, we provide several concrete examples of how cloud computing enhances the business/social environment of the client emerging nation, particularly in the areas of supply chain management, education, and health care. The field of healthcare, within cloud computing, in South Africa is examined in particular due to its need of sharing of patient data in order to improve the processes of patient transfer and to allocate health care resources optionally. An enhanced model to share patient data is present with increased capacity for the sharing of data and with improved security.**

Keywords— South Africa, patient management, cloud computing

I. INTRODUCTION

Cloud computing is generally defined as a model for providing network access, on-demand, to a shared pool of computing resources (whether hardware or software) [11] Cloud computing may include both the applications, which are delivered as services over the Internet, and the hardware/systems software that provide these services to remote clients. Cloud computing provides three new characteristics:

a) Ability to provide computing resources on demand
b) Ability to provide flexibility in the amount of computing resources required
c) Ability to pay for computing resources only as they are being used

By sharing computing resources amongst various clients, it is estimated that electricity, network bandwidth, software, and hardware costs may be decreased by a factor of 5 to 7. [1]

Based on this sharing model and its cost reduction factors, cloud computing is often referred as possessing the potential to enable emerging counties to participate in the global marketplace with minimal cost and effort. [15] Although many countries, particularly in rural areas, lack infrastructure to support cloud computing, Firdhous, in his study of factors inhibiting the adoption of IT in rural areas, discovered another factor, the non-optimal use of resources. Multiple computers would be purchased for a specific project and be idle for most of the time, outside that particular project. [5] Cloud computing offers a means to alleviate this lack of infrastructure and non-optimal use of computing resources. Infrastructure would be provided via remote facility, thus not requiring local infrastructure to be in place except for network connectivity; furthermore, computing resources would be shared among multiple users and projects in order to ensure that no computing resources are idle but that they are used to their full capacity in order to maximise their return on investment. Other advantages of cloud computing for the client include low investment in terms of pay as you go model, zero administrative cost and low IT staff requirements. Some disadvantages include reliance on the internet, security, and loss of data control. [5]

Due to its low initial cost, cloud computing is well-suited for start-ups in emerging economies, particularly in the areas of supply chain management, customer relationship management, accounting, education, and e-commerce. Combined with the pervasiveness of mobile phones with their limited resource capacity in these economies, cloud computing makes a good counter-balance with its scalable resources in order to meet Information Technology's needs. [5]

In this paper, we briefly look at several examples of how cloud computing enables emerging market nations to provide effective and feasible solutions to problems in supply chain management, education, and patient record management. This study focuses on the current patient record management system in South Africa, in part, due to the great disparity in patient record management systems between the government and private hospital organisations and the great need to integrate these records for both adequate government monitoring of health care system effectiveness and for timely access of patient information regardless of location. An enhanced model of patient record management is presented that enables patient records to be seamlessly transmitted across both the government and private

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

hospital associations and in ensuring confidentiality, legal compliance, and consistency.

II. Examples of Cloud Computing in Emerging Economies

One example of the use of cloud computing in supply-chain management is the Wang Fu Jian department store chain in China. This chain, with over 10 million customers, used cloud-based supply chain management to enable the central organisation to monitor sales in each retail store in order to determine sales trends and better establish more accurate sales forecasting, ensure adequate inventory levels, and to provide better accounting records. [8]

In the area of banking in South Africa, Nedback is using cloud computing to implement its banking operations, such as account balancing and inter-bank transfer, amongst its chain of banks. [8]

Cloud Computing has much promise in the field of education. Cloud Computing is seen as a means to alleviating some of the sub-Saharan Africa's problems such as overcrowded classrooms, inadequate textbooks, and lack of individual-teacher interaction [12; 2] through the use of online content using self-regulated learning provided through cloud computing. [2; 13; 14). This educational system allows learners to utilise remote access to cloud computing in order to proceed at their own pace through a set curriculum, be regularly assessed on their work with immediate feedback provided to them and with aggregate progress monitoring by the central agency, and to be provided with additional resources such as instructional videos and an online tutor to respond to individual student queries. [2]

In addition to online learning, efforts have been made to establish virtual computing labs, via cloud computing, to students who would otherwise not have access to these computing resources. An example, a consortium of seven universities in East Africa formed a Health Alliance in order to establish virtual computing labs which their students could access remotely via cloud computing. [8]

Using cloud computing, the Guag Dong Hospital of Traditional Chinese Medicine in China has developed a combination of data-sharing and analytical technologies, known as the Clinical and Health Records Analytics and Sharing (CHAS). Hospitals use CHAS to share electronic medical records (EMR) across their hospital network. [8]

III. Electronic Health Systems In South Africa

The Institute of Medicine defines an electronic health system as the following:

a) A collection of information for and about persons, throughout their lifetime, pertaining to their health and healthcare provided to them
b) Immediate electronic access to individual and aggregate-level information by authorised personnel only
c) Enables the improvement of the safety and efficiency of the healthcare of patients
d) Enables the support of efficient healthcare processes

[9]

In his study, Coleman outlines the divide between government and private hospitals patient data sharing in South Africa. Private hospital maintain their own data sharing network for patient records which enables the easy and quick retrieval of a patient's medical history over multiple location and the quick and accurate transfer of patient record information as the patient is transferred from department to department or from one private healthcare facility to another. In the government healthcare system, manual patient records are kept exclusively. Retrieving a manual record from the file store may take up to 15 minutes compared to nanoseconds using the electronic health record system. [4] Manual records, due in part to transcription errors, are much more prone to errors than their electronic counterparts. [19] If a patient in the government health care system is transferred from one department to another or from one facility to another, their manual patient record must travel with them. [4] This manual transportation of medical records is liable to being lost, with no possibility of a backup being available and with no audit trail of who reviewed the patient's record, when, and why.

Although a majority of South African public hospitals have instituted electronic patient records, this implementation is mostly for billing purposes rather than patient monitoring and inter-health centre exchange where manual records still predominate. [17]

In order to enable health care monitoring, the Council for Medical Schemes took into account the current state of infrastructure, data sources and needs of industry role players and developed an minimum required data set of information to be exchanged that includes demographic characteristics of the beneficiaries as well as health care access,

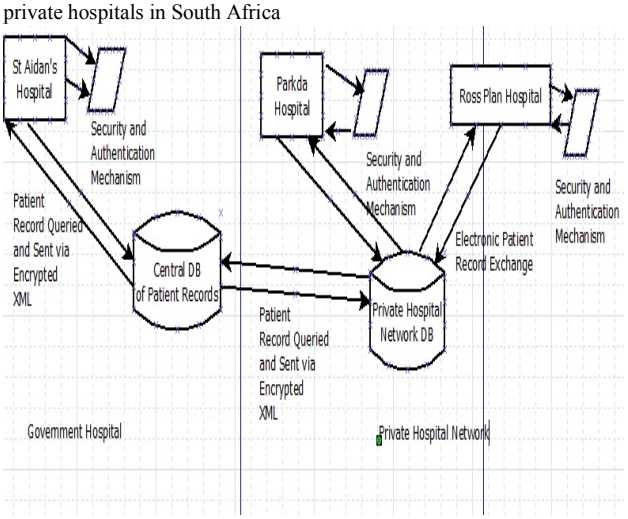
cost and utilisation of health care services. The purpose is to provide minimal inter-operability and select monitoring of performance of health care system. Additional data items, with their integration into a national schema, is an area for future research. [10] However, this minimal data set provides little data as a basis for performance monitoring and fails to address a key issue: patient mobility which could entail the transfer of life-saving particular patient information from their origin to the patient's current location.

In South Africa, the National Health Act of 2003 makes it an offence to disclose patient information without their consent, except in certain circumstances. These circumstances are "for any legitimate purpose within the ordinary course and scope of his or her duties where such access or disclosure is in the interests of the user". [6] Although health care practitioners may access this data, they must document their justification of this access. Although this requirement for justification and documentation provides needed flexibility in determining who and under what circumstances that this data may be accessed, it is very broad and only requires post-use justification in documentation as to its access. There is no standard in justification of their access. Furthermore, if multiple-person accesses are required within a short time period (such as in an emergency room situation), the documentation after-math could be quite onerous.

In addition, there is no clear guidelines as to who exactly may access this data and what authentication mechanisms may be utilised. Coleman, in his model, suggests the use of logins and passwords to authenticate and then authorise users. [4] However, this mechanism has its disadvantages, particularly in regards to impersonation and in regards to quick authentication for immediate access in dire circumstances. As an alternative, Smith outlines biometrics, such as fingerprint, as a measure for quickly accessing patient health care information in the emergency room. [18] Schneider outlines the dichotomy of HIPAA (Health Insurance Portability and Accessibility Act) in the US requiring that all patient record accesses be authorised and recorded while emergency room procedures requiring quick access with minimal interaction time. In order to meet these conflicting requirements, an emergency room in the US developed a patient record management system that authenticates users using a fingerprint scan and then determines if they are authorised to view this patient's record. After authentication and authorisation, the patient's record is quickly pulled up and displayed as long as the fingerprint-authorised personnel was nearby (distance denoted by an electronic id tag within the given proximity). If the personnel moved beyond a certain distance, the record no longer displayed. Based on the fingerprint scan and time that the id tag was within a given proximity, the personnel, time, and duration of patient record access was recorded. [16] This approach met the requirements for HIPAA's authorisation and audit trail for patient records while meeting the needs of emergency personnel for immediate access of patient records.

IV. Proposed Enhanced Model for Patient Record Management

Fig 1. Patient Record Interchange between government and private hospitals in South Africa



This model, as illustrated in Fig 1, is an enhancement of the cooperative data sharing model of Coleman. [4] The cooperative design of this model fits better into the multi-level nature of the South African hospital system than a tightly integrated design would. It has multiple components:

1) Security and Authentication Mechanism – whether based on biometrics/electronic id tags for faster and more secure access [16] or, if infeasible in some hospitals, login/password as suggested by Coleman [4], user access to patient records is authenticated and authorised at the local hospital level. In the emergency room scenario using biometrics and electronic id tags, access is authenticated and authorised but the records of this access are kept. The

personnel accessing these files must complete the justification for access at a later, more appropriate time. For auditing purposes, the details of the user access (username, justification for access, time, and record(s) accessed) is transferred with the patient record via XML as well as being recorded by the security and authentication mechanism.

2) Middleware – used by the network to handle the multi-platform and heterogeneous nature of local hospital systems in order to ensure that a homogeneous interface is provided for the central database repository. If different coding schemes are used by connected hospital networks, this middleware makes use of ontologies to translate these coding schemes into the standard WHO (World Health Organization) coding. [3] This middleware resides within an individual health care facility in order to ensure that any patient data is first translated and then sent in a homogeneous format after being encrypted.
3) Central Database Hub for Patient Records- contains patient records transferred via hospitals or hospital records using encrypted XML. Unlike Coleman whose model incorporated the messaging standard HL7 and who only suggested the potential use of XML [4], this model utilises XML to deliver data and meta-data in a standard format regardless of platform. Encryption of this patient data during transmission is used to ensure no compromising of patient record confidentiality. A standard WHO set of codes for disease/injury and treatment [20] is used to indicate diagnosis and treatment. These codes are incorporated into a patient record which also consist of national id, demographic information of patient, and cost of health care service for each associated treatment. South Africa is fortunate in that each adult individual has a unique national identification number to ensure that each individual can be uniquely identified and, based on this identification, only the correct corresponding patient record is retrieved.

V. DISCUSSION

The enhanced model illustrated above has several advantages over Coleman's model [4]. Security is improved (if biometrics/electronic id tag mechanisms are employed) and provision for auditing, a key requirement for patient access legal compliance, is enabled. Encryption of transmitted patient data ensures patient record confidentiality, another legal/ethical requirement.

XML, by its nature, allows the transfer of data regardless of the type of platform or database being used. Through the use of record structure definitions and meta tags between its data, a greater flexibility with data definitions and data transfer is permissible. [7] By using XML as the data transfer standard in this model, platform/database independence is achieved regardless of the heterogeneous nature of many hospital IT facilities within this network. Furthermore, if different fields of data are required to be sent, the flexibility of XML enables the system to quickly adapt to this change.

Standardisation of diagnosis codes, as per WHO standard, are required as per the minimal data set to be sent to the government for health care monitoring. [10] Our model expands on this standardisation of codes, requiring that treatment descriptions adapt to the WHO standard and that they be sent, as part of the patient record, to the central repository. By insisting on WHO standard, the problem of having heterogeneous treatment codes of each hospital within a patient record is reduced.

In addition, the storage of patient record data, regardless of its origin, allows a longitudal history of a patient's health to be available to health care providers, regardless of the patient's current location. This location-transparency for patient health records may be particularly advantageous in emergency situations where a patient is remote from his usual health care facility and where patient records must be both authorised and transferred quickly.

By analysing patient data stored in the central database hub, a data set of demographic characteristics, cost, and utilisation of health care services can be derived in order to provide the minimal dataset for monitoring the performance of the health care system as dictated by the Council for Medical Schemes. In addition, this database can be further analysed to reveal the prevalence and spread of certain diseases from one area to another, among other trends.

VI. CONCLUSION

Cloud computing, due in part to its low initial cost and optimal use of computing resources, offers several advantages, such as easier implementation and better system integration, for emerging nations' markets. Several examples are given where cloud computing has enabled the better provision of health care, education, and supply chain management in emerging markets. South Africa currently possesses a split between government hospitals, which largely rely on manual patient records, and private hospitals, which utilise electronic patient records. Compounding this split is the fact that most patient records that are in electronic format are used for billing rather than direct health care purposes. In order to provide a solution to South Africa's health care dilemma, we propose an enhanced model for integrated patient record management with improved security, greater flexibility, provision for legal compliance, greater standardisation and integration of data, enhanced data mobility for patients, and better capacity for analysis.

VII. REFERENCES

[1] Armbrust, Michael, O. Fox, Rean Griffith, Anthony D. Joseph, Y. Katz, Andy Konwinski, Gunho Lee et al. "Above the clouds: A Berkeley view of cloud computing.", 2009.

[2] Ally, M., R Millham, S Thakur, C Malan "Assessing a Self-Regulated E-Learning Environment for Disadvantaged Secondary Students", ICEE/ICIT Workshop, Cape Town, South Africa, 2013.

[3] Board of Health Care Providers of South Africa (BHF) "Practice Code Numbering System (PCNS)". Available at http://www.bhfglobal.com/practice-code-numbering-system-pcns. (Accessed July 10, 2014).

[4] Coleman, A "An Integrated Model to Share Patient Records in Public and Private Hospitals in South Africa", Ethno Med, vol. 7, no.2, 2013.

[5] Firdhous, M., O. Ghazali, and S. Hassan, "Cloud computing for rural ICT development: Opportunities and challenges" International Conference on Computing, Electrical and Electronics Engineering (ICCEEE), 2013, p. 680-685.

[6] Gillespie, G., 2012, "Disclosing patient records" Casebook, v. 20, 2012, p. 6-7.

[7] Houlding, Simon W. "XML—An opportunity for< meaningful> data standards in the geosciences." Computers & Geosciences, vol 27, no. 7, 2001, pp. 839-849.

[8] Kshetri, N. "Cloud computing in developing economies." Computer, vol. 43, no. 10 ,2010, pp. 47-55.

[9] Institute of Medicine (IOM). "Crossing the Quality Chasm. A New Health System for the 21st Century". Washington DC. National Academic Press. From <http://www.nap.edu/books/030907.> (Retrieved 13 March, 2008).

[10] Matshidze, P., and L. Hanmer, "Health information systems in the private health sector: pooling of resources and purchasing of health care", South African Health Review, 2007, pp. 89-102.

[11] Mell, Peter, and Tim Grance. "The NIST definition of cloud computing.", National Institute of Standards and Technology 53, no. 6, 2011, p 50.

[12] Millham, R., "Issues with Distance Education in Sub-Saharan Africa", Encyclopedia of Multimedia Technology and Networking, Second Edition, IDEA Group, 2009.

[13] Millham, R., "Issues in Mobile Learning in Ghana", Innovative Techniques in Instruction Technology, E-learning, E-assessment, and Education, Springer, 2008,p. 437-441.

[14] Millham, R., S. Thakur, and C. Malan, "Does self-regulating e-learning assist in secondary school preparation for engineering education?" Conference of the American Society for Engineering Education (ASEE Zone 1), 2014, pp. 1-5.

[15] Pina, R. A., and B. Rao, 2010, "The emergence and promise of cloud computing for under-developed societies", Proceedings of Technology Management for Global Economic Growth (PICMET), 2010, p. 1-10.

[16] Schneider, G. E-Commerce, 9th edition, Cengage, Boston, 2010.

[17] Senkubuge F and Mayosi B.M. "The state of the national health research system in South Africa", SAHR, 2013.

[18] Smith, Mark S., and Craig F. Feied. "The next-generation emergency department." Annals of emergency medicine, vol. 32, no. 1,1998, pp. 65-74.

[19] Suomi R "Introducing Electronic Patient Records to Hospitals: Innovation Adoption Paths". Ideal Group Inc, USA, 2001.

[20] World Health Organization. International classification of procedures in medicine. Vol. 1. World Health Organization, 1978.

Big Data Infrastructure for Aviation Data Analytics

Anandavel Murugan[2], Dinkar Mylaraswamy[1,*], Brian Xu[1] and Paul Dietrich[1]

[1]Honeywell Aerospace, Golden Valley, MN, USA

[2]Honeywell Technology Solutions Limited, Madurai, India

Abstract. *This paper describes our approach towards developing and using a big data infrastructure for analyzing aviation data. In this paper, we briefly introduce our data sources, nature of data collected, cluster design, data loading and storage strategy and our language and library of choice for analytics and visualization. We present some of the analytics we have implemented for health monitoring of auxiliary power units (APUs) using our big data infrastructure. Best practices to implement big data analytics and pitfalls are discussed and substantiated with our experiences.*

Keywords—big data, analytics, algorithms, aviation, cluster, hadoop

I. INTRODUCTION

Recently every industry is witnessing a dramatic increase in amount of data being collected. With the advent of the parallel and distributed computing paradigm, analyzing big data has become practical. Many industries including aviation industries are starting to utilize big data and analytics for gaining competitive advantages. Though it is not new for aerospace companies to provide flight data solutions, their offerings have expanded with this sudden increase in data being collected. Legacy aircrafts used to capture 125+ flight parameters, but Boeing 787 captures more than 1000 flight parameters [1] with some reports claiming half a terabyte of data per flight. This explains the big data explosion in aviation. Apart from these flight data, a large amount of data get generated in repair shops, inventory systems and by various regulatory organizations as well. Analyzing such big data can help in improving flight safety, reducing operational delay, better inventory management of spares, predictive maintenance of various equipments on board, improving fuel economy of the fleet etc. Honeywell Aerospace has been a pioneer in providing flight data solutions for predictive maintenance of parts through its tools and data sources like PTMD (Predictive Trend Monitoring and Diagnostics). Our focus is to leverage latest tools and technologies in big data analytics space to improve these predictive maintenance offerings. Using big data and analytics infrastructure, we have enabled analysis of large data from ACMS (Aircraft condition monitoring systems) and various repair databases and found outliers and hidden trends in the data.

In this paper, we primarily introduce our big data analytics stack and substantiate it with a case study. This paper is organized as follows: section 2 shows the data source, section 3 presents our big data stack with each sub-section explaining various parts of the stack, and section 4 demonstrates one of the solutions we have built for Auxiliary Power Unit (APU) analytics using the big data.

II. AVIATION DATA SOURCES

A. Operation / Field Data

Digital flight data acquisition unit (DFDAU) is the primary source of flight operational data. It is a mandatory electronic device that should be installed in every aircraft. It collects and preprocesses discrete, analog and digital signals captured from multiple sensors and various avionics systems in the aircraft. The processed data is passed to the DFDR (Digital flight data recorder) that provides magnetic data storage for the captured data. Over the days, DFDR has evolved and the latest QAR which stands for Quick Access Recorder allows using removable storage devices and also wireless transmission of the stored data. Current DFDAU also includes ACMS (Aircraft Condition Monitoring Systems). This system records data from multiple sensors and it is mainly used to monitor the health of various components in aircraft like engines, APUs, brakes etc. ACMS captures the data in multiple scenarios. Snapshot data are captured during key phases of flight like take-off, cruise and touch-down. Summary data are captured and recorded during end of flight. It also captures the data during abnormal conditions like APU shut down, higher turbine gas temperature during take-off etc. Typical parameters captured include aircraft speed, altitude, position, rotor speeds, exhaust temperature, various pressures, vibration etc. All these data are captured in different report formats. This report format can vary between DFDR manufacturers and also between aircraft equipment model. A key challenge in handling the data is to accommodate these multiple report formats.

B. Repair Data

Repair shops are another big data source in the context of predictive maintenance. Repair shop investigations when correlated with operational data can provide useful insights to reduce recurring equipment problems. Repair reports are mostly free form text captured by repair shop personnel as part of their investigation and repair. Some of the reports also include the observation made by the pilot or crew when the event occurred. Such repair data are mostly unstructured that need the application of complex processing to derive meaningful inferences.

III. BIG DATA INFRASTRUCTURE

A typical big data stack includes a distributed computing infrastructure, ETL layer to load the data, analytics module to

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

process the data, web services layer for the users to consume the analytics and interactive visualization for reporting needs. Though many big data vendors claim to provide a complete solution, from our experience we found that there is no single recipe for implementing a big data solution. Instead it has to be built, by leveraging best in class tools from multiple sources. We have built our big data infrastructure by combining multiple open source software like Hadoop, HBase, Shiny server, open CPU server and languages like R and Python. Our cluster is built using Linux machines running SUSE Linux. Users consume the analytics mainly through a browser, while some of the users use R and MATLAB.

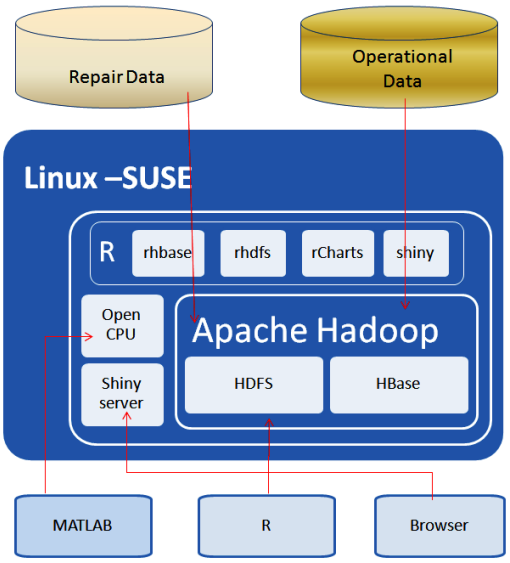


Fig. 1. Architecture of our Big data infrastructure.

A. *Apache Hadoop*

Our big data infrastructure is created on top of Apache Hadoop (http://hadoop.apache.org/). Apache Hadoop is a distributed computing framework which provides large scale data processing on top of cheap commodity hardware. It mainly comprises a distributed fault tolerant file system called HDFS (Hadoop Distributed File System) and a resource management platform called YARN. It supports map reduce programming model which allows multiple map and reduce tasks to operate on different chunks of data and run in parallel enabling faster execution. Apache Hadoop acts as a backbone with related tools like Apache Hbase, Apache HIVE, Apache Pig and Apache Spark aiding in different aspects of big data processing. We use HDFS for storing the raw reports, HBase for storing the metadata of reports and Hive for running SQL queries.

B. *Apache HBase*

Apache HBase is a columnar database built on top of HDFS. It lets the data to be stored in a way convenient to query rather than in normalized form as in relational databases. Its columnar nature helps in accommodating schema changes without much problem. In HBase, data are stored in tables and each table can contain one or more column families with each of the column families containing one or more columns. The column families should be fixed during table creation, but the number of columns in each family can be added as and when needed. This makes HBase flexible to accommodate the schema changes. In our big data stack, we have used HBase as a metadata store for various reports. Each report is generally identified by asset/part serial number, aircraft tail number, airliner and it has a specific timestamp. Along with these metadata, the HDFS location of the actual report file is also stored in the HBase table.

HBase tables are normally designed for query performance. To have better scan performance, we went with an approach of having separate tables for each report type. For row keys, we decided to have a composite key with asset model, serial and operator details as part of the key. This is in accordance with querying pattern. This helped us in using row key filters while scanning or querying the table. Row key filters are very efficient when scanning the table, as it scans the row keys only. Though composite keys are good for scan performance, they can lead to a phenomenon called region server hotspotting. It is a case when more data are accumulated in one region server when the row keys are alphabetically closer. This overloads one region server when the other region server is underutilized. To avoid this, random characters are appended to the row key to make the data to be distributed in multiple region servers

C. *Hadoop Distributed File System (HDFS)*

HDFS is the distributed file system built on top of Hadoop. We store the reports of each asset in HDFS and a path identifier to locate the file is stored in HBase. Apart from scalability and availability considerations, the key reason for storing the files in HDFS is to run map reduce jobs on those files. HDFS can store the files in any format and in any way. We considered a few aspects while arriving at the storage strategy. Reports need to be logically grouped, based on typical access pattern. Organization of reports should support efficient map reduce jobs. Since our report size is generally less than HDFS block size (e.g., 64 MB), we decided to consolidate the reports into map files which favors other considerations as well. A map file is a native Hadoop file format built of top another Hadoop native file format called sequence file. While contents of sequence file cannot be looked up by key, map file could be. All reports of an asset are stored as byte array in a map file with keys matching the path identifier stored in HBase.

D. *R – Platform for Statistical Computing*

Analytics, big or small involves statistical computing. It might be as simple as calculating an average or as complex as running an unsupervised clustering of multi dimensional data. Though Apache Hadoop prefers Java as the language for authoring map-reduce programs, it is not a language of choice for mathematical calculations. Authoring a statistical analysis in a java map reduce program would be verbose. Hive and PIG which are other popular alternatives are very easy to program, but are not expressive enough for a statistical analysis. So we have selected and used R as the language for

our statistical analysis. R is a cross platform programming language and environment for statistical computing (http://cran.r-project.org/). From the latest survey results, it is the most commonly used language among data analysts. A vast array of packages is the key strength of R. As of today, CRAN which stands for Compressive R Archive Network hosts 5742 R packages. CRAN has mature R packages for various analytics like statistical analysis, machine learning, text analytics and visualization. Adding to the existing packages, latest packages like RHadoop and RHIPE help in implementing R based analytics which works with Apache Hadoop.

E. Data migration (Extract Transform Load Scripts)

ETL (extract, transform and load) is the term used to denote data migration between multiple systems. In the ETL process, data is extracted from one source, transformed into some other form and stored in another destination. Hadoop requires the data to be stored in HDFS for the map-reduce jobs to operate on, but storing the data in HDFS involves moving the data from its parent data source. There are a few apache projects built for this specific need. For example, Apache Flume is used for aggregating data from multiple agents into HDFS. Agents could be log files, JMS, directories and even sensors. Apache Sqoop is another project which is specifically for moving bulk data between Hadoop and structured relational databases. Since these are built for broader scope, it does not fit specific needs of our implementation. So we had to create our ETL scripts completely from scratch. Our ETL scripts are authored in Python using some of the Python Hadoop modules like HappyBase and Pydoop. While Pydoop allows accessing HDFS, HappyBase enables HBase communication through thrift interface. From the data sources, we extract the data and load the data into HDFS and HBase. Our ETL script has a nice feature to do a dry run of data migration. During that, only a log gets generated and none of the data gets loaded into HDFS or HBase

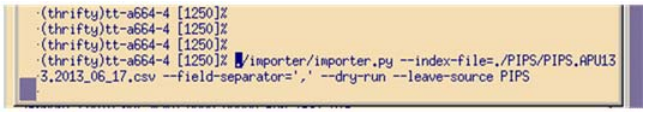


Fig. 2. A Screenshot of ETL script importing data into HBase and HDFS

F. Analytics as Service

Not all users of the data create analytics. In most cases, only a few people get involved in creating analytics while the rest use the analytics for making their business or engineering decisions. The web has become the friendliest interface for consuming any data and using a browser for consuming analytics has multiple advantages. It can be accessed from any hand-held devices like tablets and smartphones. But there are some other users who prefer the "true" web services i.e. not websites. Web services are ideal as they provide platform and language interoperability. To cater to these two needs, we have used two deployment models. We have used Shiny server as deployment server for the websites created on top of analytics. For the analytics to be exposed as web services we have used Open CPU server which exposes any R package as a REST web service.

1) Shiny Server and Shiny R package

Shiny is an R package that helps in creating R enabled websites with limited web development experience. Using this package, R developers can quickly turn their analytics into an interactive website with a few simple steps. Every Shiny application has two R scripts, ui.R and server.R. ui.R is supposed to contain user interface controls laid out in suitable layout. server.R is supposed to contain the R based analytics to be embedded in the Shiny application. Though it is recommended to have this separation, it is not a strict rule. So server.R can host user interface widgets and ui.R can contain analytics too. Shiny uses the reactive programming model. Reactive programming is built around data flows and propagation of change. In simple terms, one state change triggers reactions which flow down and create further state change. In the case of a Shiny application, a reactive model is enabled by connecting a reactive source with a reactive end point. Reactive source can any be any widget whose state changes because of user interaction and reactive end point can be a plot or table or simple text which is dependent on the state of the reactive end point. For example, based on selection of an entry in the drop down list, a plot can be re-rendered or a table be can rendered. In this case, drop down box is the reactive source and plot and table are the reactive end points. Shiny package contains a lot of user interface widgets which are needed for a complete application. But if need arises, any HTML or Javascript controls can be used easily. Shiny also provides some predefined layouts which can be used to lay the widgets in the user interface. While Shiny package is like a SDK used to create the Shiny application, Shiny server is the deployment environment to host the Shiny application on the web. Just like any SDK which provides an execution environment for development and testing, Shiny package also has a basic server. But it can host just one application at a time and it should be used for development and testing only. Shiny server is useful for hosting multiple Shiny applications.

2) Open CPU

Open CPU is an R package which exposes analytics capability of R as RESTful webservice. With this HTTP API as back end, any HTML based applications can be built easily. In contrast to Shiny which does not need any web development experience, building applications for open CPU needs HTML/javascript programming experience. Just like Shiny applications which are hosted in Shiny server, open CPU applications are hosted in Open CPU server. Open CPU R package also contains an embedded web server for development and testing. Since Open CPU API is RESTful, even applications which are not browser based can utilize them easily. So Open CPU can be used to build not only websites, but also integrated solutions.

G. *Visualization using rCharts*

Visualization is a key ingredient in analytics. Interactive visualization reduces half of the analytics' burden. Traditionally to visualize characteristics in data, plots like histograms, bar charts, scatter plots, box plots were used. But with the advancements in Javascript and CSS, graphs and plots have become highly interactive and rich in content. Hence information displayed in multiple plots is easily condensed into a single plot by letting the users to zoom in/out or turn off a few variables or change scale of axis. Since our deployment is on web, we have used javascript based plots to provide interactivity. We have chosen rcharts, an R package as our charting solution. It follows familiar lattice R package style commands. Currently it supports many Javascript libraries like dimple, nvd3, polycharts, xcharts, rickshaw etc and any new javascript library can be easily integrated as rCharts is designed as a Meta framework. Though rCharts is available in business friendly license, the underlying javascript libraries may be distributed with a specific license. So caution needs to be exercised in choosing the right underlying library. Another R package used is ggvis which is similar in theory to famous ggplot. It uses Shiny's reactive programming model and dplyr style for data manipulation.

H. *R Studio*

R Studio is an open source integrated development environment for R. R itself comes up with a minimal development environment, but it is not easy to develop a complete R package or a complex Shiny application using just R terminal. R Studio provides easy project set up templates, visual debugging, code completion, roxygen comments, integrated help etc which makes development fast and error free. Another notable IDE option is StatET plug-in for Eclipse for people who prefer Eclipse.

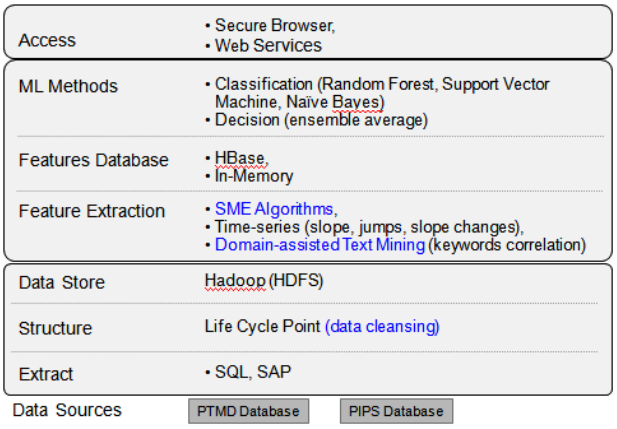


Fig. 3. Another perspective of our big data stack

IV. AUXILLARY POWER UNIT LIFECYCLE ANALYSIS

This case study explains how big data enabled the life cycle analysis of some Honeywell made APU used by multiple airliners. Analytics is created using the big data infrastructure and hosted as a Shiny application.

A. *Auxillary Power Unit*

APU is a small gas turbo engine which provides electrical and pneumatic power for an aircraft. It is generally used when the aircraft is on ground before the engine is started. It provides bleed air for starting the main engine and electrical power to the whole aircraft. Some of the bleed air is also used in environment control system for cabin pressurization and for cabin cooling/heating

Honeywell aerospace has a program called PTMD (Predictive Trend Monitoring and Diagnostics) which provides predictive maintenance alerts about the APU being used by the specific airliner. It is a subscription based service which uses the data captured by the sensors in the aircraft. PTMD gets the data through Aircraft Communication and Addressing Reporting System (ACARS). This data is available in multiple report formats based on the model of the APU. We have loaded part of this data into our cluster using our ETL scripts.

Either because of the PTMD alerts or because of the faulty conditions observed, the APU is sent for repair. Honeywell Aerospace has a repair shop database called Product In-service Performance System (PIPS) which captures the symptoms as observed by the pilot or crew, observations of the repair technician, repairs done etc. Except for the complexity in handling the free form text in this database, PIPS data is very straight forward to use. PIPS data extracts of few APU models are loaded into HBase using the ETL scripts.

Though multiple algorithms were created using PTMD data and PIPS data, without the big data infrastructure, they were not analyzed in a combined way. APU life cycle analysis is one of the interesting analytics we have created using PTMD and PIPS data in our big data infrastructure. Life cycle point (LCP) of the APU is the time interval between two successive repair shop visits. LCP analysis is crucial as it helps in analyzing the operational characteristics of the APU between any two repair visits. Even an analytic as simple as calculating the life cycle points of different APU models used by different airliners will give valuable insights into the repair and operating trend of the APU

B. *Life cycle point calculation*

Our ETL script populates the PIPS and PTMD table from data extracts. LCP can be calculated by identifying dates of all the repairs and finding the operational data between the successive repair events. In every PTMD report, along with other parameters, APU start time is also stored. PTMD report parsers return the parameter values either as an array of numbers or strings. We identify the index of APU start time which falls right after the first repair event and last index of APU start time which precedes the second repair event of the LCP. These two indices are enough to get the values of any operational parameter like Exhaust Gas Temperature (EGT) for this LCP. We have R package which calculates the LCP using this logic and stores indices along with other LCP

details in another HBase table, avoiding recalculation. We have R Shiny application which acts as front end for the users offering multiple analytics on top of this LCP data

C. *APU usage trend analysis*

Each APU might be used in different patterns based on the airliner and route being served. Accordingly the repair trend might also be different. An analysis was done to study the usage trend along with repair trend. For a specific APU model, using the repair event intervals, average usage per day is calculated from APU usage hour parameter which is captured in PTMD reports. From the LCP table, number of life cycle points of a specific APU model used by multiple operators is retrieved. When the number of APU being used, number of LCP and average usage per day are plotted together, it gives an important inference. Operator A is using more APUs and hence has more repairs. But still the average usage per day is much less when compared to others. This is little anomaly which can be closely scrutinized further.

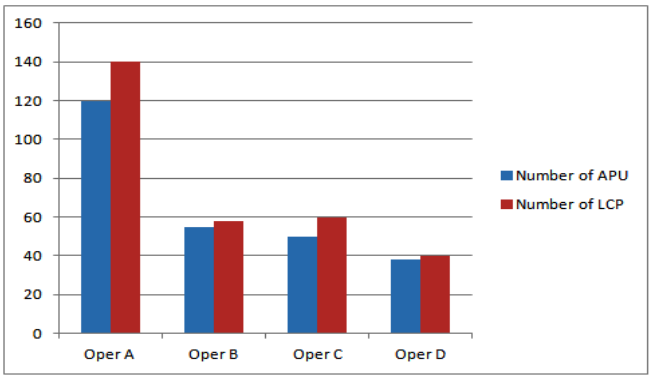


Fig. 4. APU repair trend

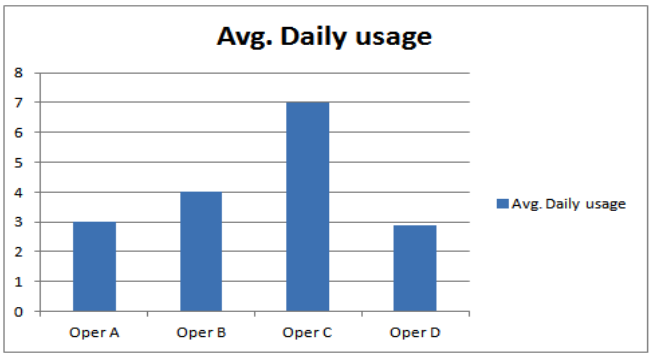


Fig. 5. APU daily average usage trend

D. *Exhaust Gas Temperature analysis*

Exhaust Gas Temperature is a measure of thermodynamic efficiency of an APU. APU is supposed to convert the thermal energy created by burning of fuel into useful work. If all thermal energy is utilized for useful work, amount of thermal energy let off in exhaust should be lower and hence EGT should also be lower. If the EGT is higher, it means APU is not efficient and it needs to be repaired.

Another analytic allows visualizing the trend of any specific field parameter during a specific LCP. This helps in visualizing how EGT has varied during the life cycle and spot outliers.

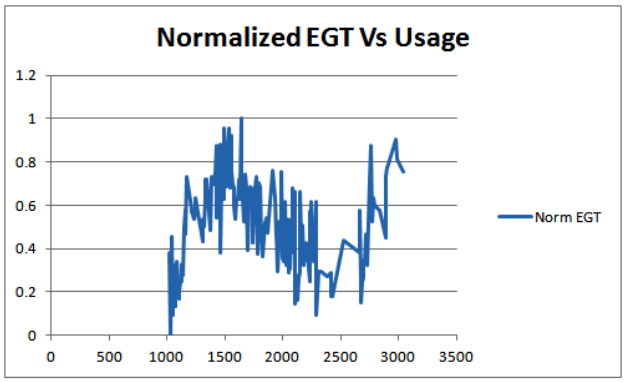


Fig. 6. EGT Trend

E. *EGT Margin distribution*

EGT margin is the difference between maximum allowable EGT in the ISA (International standard atmosphere) condition and actual EGT achieved. As the APU ages due to compressor & turbine damage or because of accumulation of dirt on the compressor blades, EGT margin drops. When the EGT margin falls below a predefined limit, it is sent for repair. Using the LCP data along with field data, we have analyzed the distribution of EGT margin across airliners. This analysis helps in finding the airliners whose EGT margin has unusual distribution across repairs. EGT margin is a field parameter, but when it is analyzed across repairs, it helps in enhancing predictive analytics.

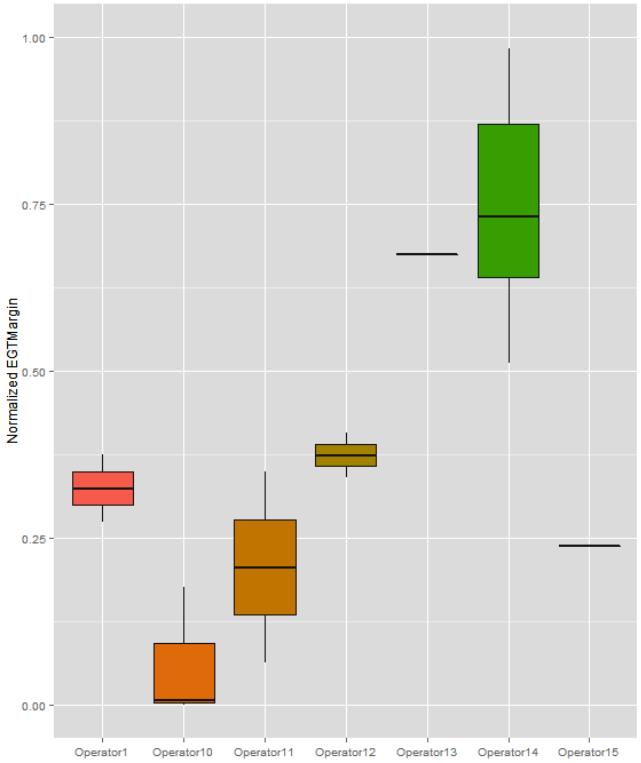


Fig. 7. Box plot of normalized EGT margin of some operators

The case studies discussed here explain how new and novel perspectives could be easily developed by employing big data. In our case, columnar nature of HBase which is the important part of our big data stack enabled us in storing data of different schemas in one database. R along with hadoop packages like rhbase and visualization packages like ggplot, rCharts enabled querying and plotting the data with ease.

V. CONCLUSION

In this paper, we explained our approach for using big data tools and provided a preview of its usage using a single case study. We explained all the open source software we had used and also discussed how they interact with each other to provide a complete big data environment. We also explained the nature of the data sources and how it affected the storage strategy. We also explained how big data can help in bringing two different data sources together and enable holistic analysis, using our APU life cycle analysis case study. Though the case study we had chosen to highlight is not very complex, it introduces the need for big data.

APU is widely discussed in this paper, but we are slowly extending the big data analytics to other aircraft assets like brakes too. Our future focus is to embark on unsupervised analysis using advanced machine learning algorithms.

References

[1] Neil A.H. Campbell. The Evolution of Flight Data Analysis. ASASI, 2007

[2] B. Xu, D. Mylaraswamy, and P. Dietrich. A Cloud Computing Framework with Machine Learning Algorithms for Industrial Applications. WorldCom ICAI, July 2013, Las Vegas, USA

[3] B. Xu, D. Mylaraswamy, P. Dietrich and Anandavel Murugan. Case studies: Big Data Analytics for Health Monitoring. WorldCom ICAI, July 2014, Las Vegas, USA

[4] HappyBase, http://happybase.readthedocs.org

[5] Pydoop, http://pydoop.sourceforge.net/docs/

[6] rCharts, http://rcharts.io/

[7] PTMD: https://commerce.honeywell.com/webapp/wcs/stores/servlet/eSystemDisplay?catalogId=10201&storeId=10651&categoryId=42999&langId=-2

Client Requirement Modeling using Resource Broker Architecture in Cloud Computing Environment

Chetan Awasthi[1], Priyesh Kanungo[2]
School of Computer Science & IT,
Devi Ahilya University,
Indore, India
[1]chetan_awasthi1@yahoo.com, [2]priyeshkanungo@hotmail.com

***Abstract*— The Cloud has become platform to provide variety of services and a broker is needed for quality service at competitive rate and to help the vendors in increasing revenue by keeping their resources. In this paper, we have proposed a resource broker mechanism that will help the vendors to identify frequent clients (FC) and provide quality services thereby improving trust on client side. The broker mechanism helps the clients and vendors in locating, resources and increasing their revenue respectively.**

Keywords—Frequent Client; Broker; cloud computing;

I. INTRODUCTION

Cloud Computing is the emerging field that made huge computing power easily available for every organization. It describes a novel supplement, utilization and delivery paradigm for IT services over the net. Technology offers the computational power, bandwidth, storage, software usage, software development and testing etc. The cost effectiveness of cloud technology is achieved by providing 'pay-per-use' model and professional management of the infrastructure. The cloud computing provides various services in form of XaaS (i.e. everything as service) e.g. Infrastructure as a Service (IaaS), Platform as a Service (Paas), Software as a Service (SaaS) etc.

Software as a Service (SaaS) refers to software's which are available on rent over the internet or in some other mode for user. Therefore, the user doesn't have to worry about the installation, licensing, up gradation etc. He simply uses the required software.

Platform as a Service (PaaS) i.e. platform are also available on rent over the internet or in some other mode of services. Here platform means operating system like windows, Linux, Unix, etc. The user simply login to the required platform.

Infrastructure as Service (IaaS) is available to users which are professionally managed and not otherwise possible for every user. By infrastructure we means Processing Power, Memory, and Storage, etc. Moreover air condition and power conditioning equipment are not required as the servers and storage devices are maintained by the service providers.

A number of computing services like Windows Azure, Amazon EC2, SalesForce.com, Google etc are available that provide different service to the client. These services are available on rental basis. Due to a large number of cloud service providers, it is difficult for the clients to choose right vendor for completing their tasks. From here new term emerges which is popularly known as resource broker. The broker on behalf of client, search vendors', required services or resource to help the client.

At broker side also, various clients are available who demand the same resource or at same time broker is busy in fulfilling the demand of some different client. Thus, there is need for appropriate allocation mechanism. Different researchers have proposed resource allocation mechanisms for the allocation of resource or service to the clients at resource broker level. In this paper, a mechanism is proposed for broker which helps him in allocation of resources or services to various client. The proposed mechanism works on the basis of frequency of user. i.e. user who visited to broker more frequently, who are given priority in allocation of resource. We called such user as Frequent Clients (FC). The benefit of using proposed mechanism is that, the reliability factor of FC increases. Since they are specially treated while job allocation. They have to wait for less time in comparison to the user who did not visit to same broker frequently. As compared to other brokers, generally uses FCFC method for resource allocation. In such scenario, users who are frequent and provide business to the broker most of the time have to wait. So there may be chance of losing same type of customers. The proposed mechanism also uses the concept of FCFS whenever two or more FC has same frequency.

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

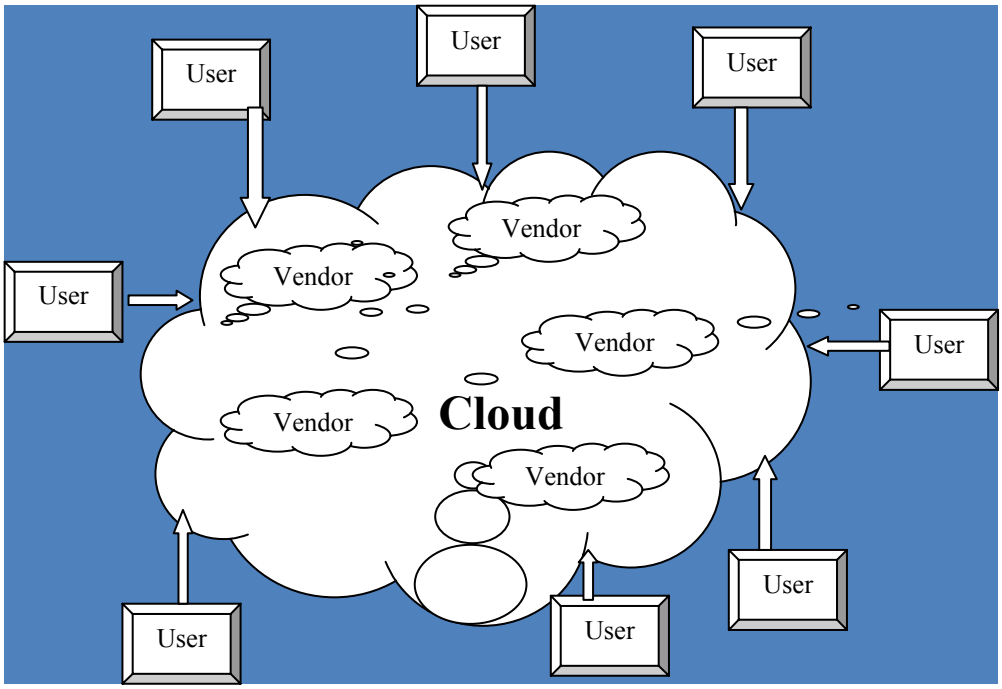


Fig. 1: Basic model of Cloud computing with cloud service provider and the user

The average turnaround time of user also decreases in this system. Broker also keeps track record of user which in turn help him in finding and serving the best service to his client. In future, broker may keep advance reservation of resource for such user with the help data mining requirements which clients submitted with initial request.

CloudSim-3.0.3 toolkit is used to test the proposed algorithm which is simulation software for cloud developed as a part of cloudbus project at the University of Melbourne, Australia. It provides modeling for power aware, energy efficient computing. It also has the feature of modeling internal datacenter topologies and message passing applications [1].

II. Literature Review

The diverse requirement of client and various services provided by the cloud computing vendors have given birth to the concept of cloud computing resource broker. The main task of such broker is to match the service requirement of the client with the different service provided by the various vendors. While matching, the request broker also keep track of the parameter that may or were given by the client e.g. price, time, resource list etc. In this manner, cloud computing resource broker helps the client in searching various services at best price and in less time. Moreover, it helps the vendors by keeping their resource busy for most of the time, which in turn increase their revenue.

The broker who acts as middle layer between vendor and user [2] makes agreement for the services between them. The author has explored the use of broker mechanism for user in search of services provided by the vendors. The broker takes users requirement as input and then matches required service with the services provided by the vendors.

The cloud broker maintains Service Level Agreement (SLA) between various cloud services providers and the clients [8]. In this case, an organization approaches the broker with a computing requirement and while matching the services from providers list, broker also keeps in tracking of SLA agreement that were set up between the two parties. The secure cloud broker architecture should also address to following points: -

(i) Data confidentiality problems and effective matching of requirements of the client with service provider's list of services. (ii) Performance check on SLA and action against its violation. (iii) Analyze of risk and take appropriate action if required.

In a large organization, it is difficult to manage internal resources due to large number of users' and their service requests. In a cloud computing environment broker acts as manager who tracks users' request and allocation of resource from the cloud datacenter. In this paper meta-broker concept has been proposed for inter-cloud. In these concepts broker coordinates cross exchange and service automation transparently. The result of meta-broker shows the performance level of average execution time for different users who submit their requirement concurrently [3].

The use of SLA for brokering of services provided by various vendors [5] was discussed. By the use of SLAs broker gives opportunity to vendors to keep their resource information internal and secure and at the same time provide the users guarantee on required service submitted. The proposed model keeps state of resource private.

In present era, computer technology plays vital role in medical science field. To enhance the service of computer and information in medical field cloud computing cam be deployed. The goal-based cloud broker system proposed by

the author provides solution for retrieving data from the index metadata [4]. The user send request to goal-based broker who discover the resource from the index server. Here index server is again stored in index form and the brokers keep track of it. The result of search provides fast discovery and selection fast which was the author research objective.

Generally, in a business model of cloud computing users pay for the resource or the service they have used. The author proposed agent based testbed for the searching of cloud resource and SLA negotiation between provider and clients [6]. The work had been carried out in four stage resource discovery process. These steps are selection evaluation, filtering, and recommendation. With these broker agents match the request of client with the services published by the providers. The result shows the success of matching request and services submitted and published by client and providers respectively.

Cluster Computing, Grid Computing, and Cloud Computing are the environment which is commercially accepted today. Scheduling of resource and process in this environment is difficult and challenging task. The author in this paper makes scheduling algorithm and design policy which can be used for distributed environment mention above [9]. The main aim of this paper was load balancing between cluster to grid, modification in the existing load balancing policies, analysis to find extreme point of existing dynamic content caching and to point out the issues like scalability, connectivity, resource discovery etc in grid computing environment.

Nowadays, business processes and their architectural plans are changing rapidly to meet the market requirements. This ensures that systems are up-to-date and running on the latest business processes which organization plans [7]. The change management process that can back up the business process which are running cloud computing were proposed. The task performed by cloud broker here are discovery and binding process model for the initial time setup, monitoring business process changes, identifying non-explicit changes which are fired by process changes, and carrying out the changes while service binding.

Various application services were proposed for the provisioning the complex task composition, configuration and deployment requirements. It is difficult to evaluate the performance among these different application service available for cloud computing. The author presented a CloudSim which is extensible simulation toolkit and enables modeling and simulation of cloud computing system and application simulation environment [1]. This toolkit support modeling and simulation of data centers, virtual machines, and resource provisioning policies. It is being used by several researchers for simulating their work for cloud computing resource provisioning and for the management of energy efficient data centers.

In the above literature review, various brokers and brokering mechanism have been studied. We find that none of the above mention broker identifies the point of allocation of resource on the basis of frequency of user visits. This paper based on the concept of giving priority to the user whose frequency of visiting to broker is maximum.

III. Informal Description of the Algorithm

Broker registers the client for various information like the business resources required, financial turn over, time for which user had been in the market etc,. This information helps broker in finding future trends of services he may interest in. Generally, broker works on the basis of demand submitted by the client. The broker searches for the best option from the market. The search results save time and cost of client in finding best service at competitive rate. In this way, pay-as-we- go model is provided to the client by the broker.

In this paper, broker mechanism work on the basis of finding Frequent Client (FC) and allocate the resources to him first so on. Frequent Client is one who visited to broker for maximum times for his resource requirement than any other client. Thus broker counts the repeated user who visited at him most of the times. Broker provides priority in allocation resource to FC since they are regular customer for him.

In figure 2, U1, U2, U3 are clients who submit their requests to broker in the form of required resource plus Unique ID of each user (R.R. + ID). From here, the broker processes the request in two phases. In first phase consists of processing, broker digs out the ID of each user and check it with the list of ID already stored in its database and increases its frequency count by one. However, if no match found then the frequency count of that ID is set as one. This will be incremented next time when the client with same ID visits again. Then, ID with maximum frequency count are passed to resource scheduler. In phase two, resource is allocated to that ID and allocated resource ID by resource scheduler is kept for further use.

The steps involved in allocation resources to FC client include:

To begin with the client is registered at the broker database.

Broker should also have registered various vendors and the services provided by them which available in the market.

If the client registered for the first time and submits requirement, then the broker stores types of requirement submitted and generates a unique ID for the client and provide service to him. Every client should give required ID or registered himself for getting new ID.

If the same client visits frequently, then broker gives priority to him during allocation, that client gets the required service easily, in peak time, or at low prices. It also increases revenue of the vendors and brokerage fee for broker. Broker can give relaxation to such FC in brokerage fees.

To find frequently visiting client

Add every visit of every client $\sum(f_{i=1..n})$. *Now search for the client whose frequency is maximum* (U_{id} *with Max F) and if he is the one who requested for resource then priority is given to him. After this, next FC will get the chance.*

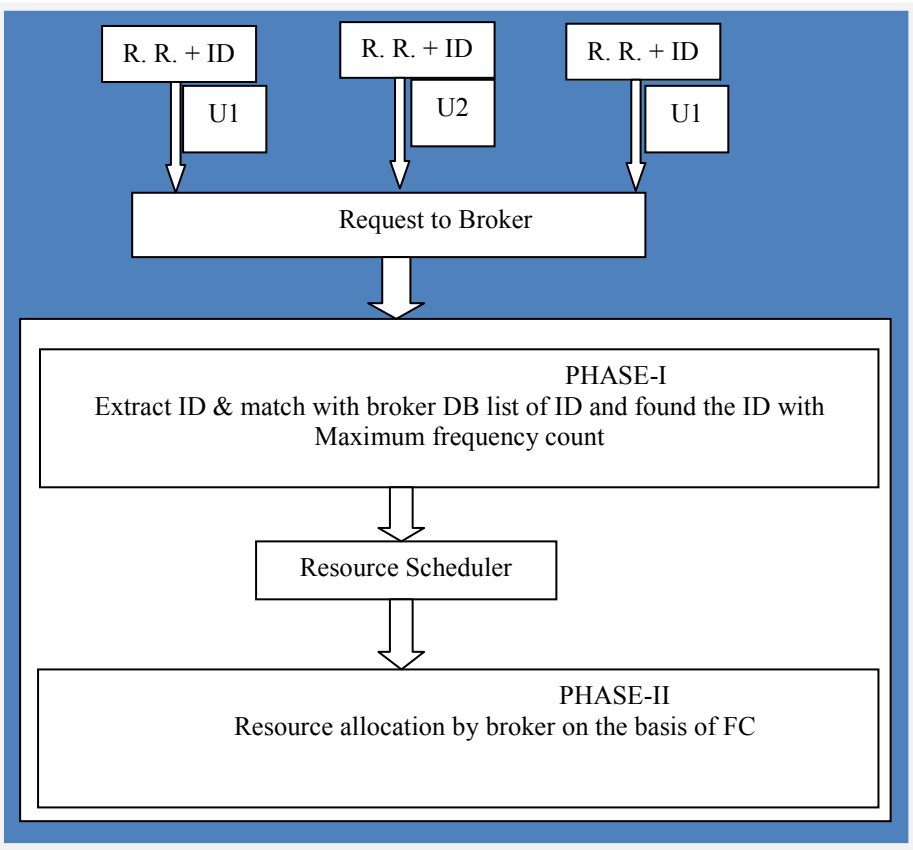


Fig: 2: Architecture of the Resource Broker in a Cloud Computing Environment

The above mentioned mechanism can be implemented on daily, weekly, yearly basis.

Once the client is known as FC, the broker provides various offers and scheme to this category of clients.

There will difference in brokerage fee for normal user and FC as the FC users generate more business for broker than the normal user.

IV. Formal Description of the Algorithm

1. Register the Client at the Broker DataBase (DB).

2. Broker also registers different vendors and the services provided by the vendors.

2. IF new Client: ID generated and submitted resource requirement (RR). This will be store in Broker DB

ELSE

ID and RR are submitted

3. Broker searches for required resources on the basis of parameters submitted by vendors. Search in broker-resource list

IF resource stored in advance by the broker, allocate the resource (s)

ELSE

Search the vendors who provide the required resource

4. Broker monitors frequently used resources; hire the resource from vendor in advance

5. Repeat step 2 through step 4

6. End.

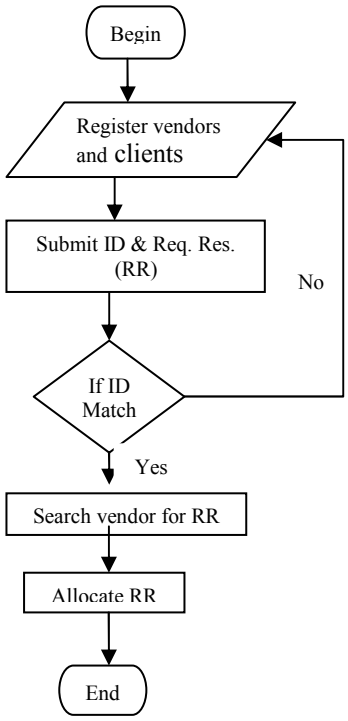


Fig. 3: Flowchart of the algorithm

V. Simulation and Analysis of Results

To test the proposed algorithm, a simulation toolkit labeled "CloudSim" is used [CloudSim: a toolkit for modeling and simulation of cloud]. It is a generalized toolkit for modeling and simulation. It also allows performing experiments on evolving cloud computing infrastructure and application services [1]. This experiment was carried out in controlled manner. The major advantage of using toolkit is time effectiveness, flexibility and applicability.

The results of simulation are shown in Table 2 and Table 3 and Figure 4 and Figure 5. The result Table 2 shows the waiting time on the basis of proposed work and in the Table 3 waiting time shows on the basis of FCFS. The graph in fig.4 shows that the client with high frequency count for resource request has less waiting. The graph in fig.5 shows the frequency count and waiting time of the clients using FCFS technique.

Fig. 4 compares the waiting time and frequency of the client using priority allocation on the basis of frequency count of the client. Thus we say that the proposed method gives better service to the clients with higher frequency of resource request.

On the basis of above results following graphs were plotted. In the first graph proposed work is seen. In second graph general allocation on the basis of FCFS is shown. From this it is clear that if broker apply the proposed work then average time of client can be decrease. And also the client who are most frequent visitor to broker can be given preference. This will also increase trust worthiness between broker and client.

The Virtual Machine (VM) used by broker in proposed work has following attributes:

ID (for VM)	0
Mips	250
Size (in mb)	10000
ram (in mb)	512
bw (bandwidth)	1000
pesNumber	1 (no. of cpu)
Vmm	Xen(vm name)
Vm-scheduling	Timeshared

Table 1

User ID	2	5	6	1	4	3	7	9	8	11	44	22	77	31	98	100	26
Frequency	3	1	7	4	4	2	4	1	1	9	1	2	10	3	4	6	7
Wait Time	5	7	3	4	4	6	4	7	7	2	7	6	1	5	4	4	3

Table-2

User	2	5	6	1	4	3	7	9	8	11	44	22	77	31	98	100	26
Frequency	3	1	7	4	4	2	4	1	1	9	1	2	10	3	4	6	7
Wait Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

Table-3

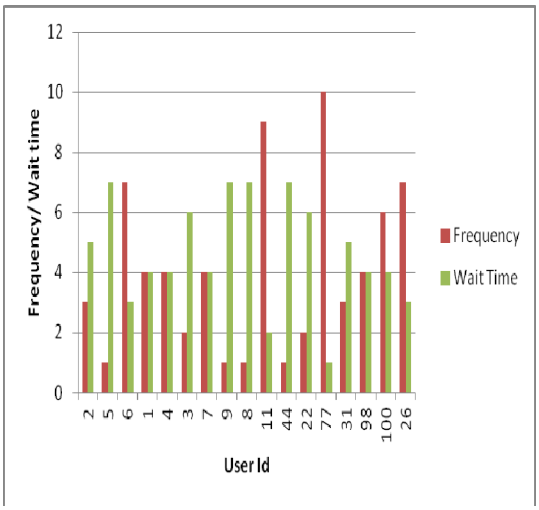


Figure 4

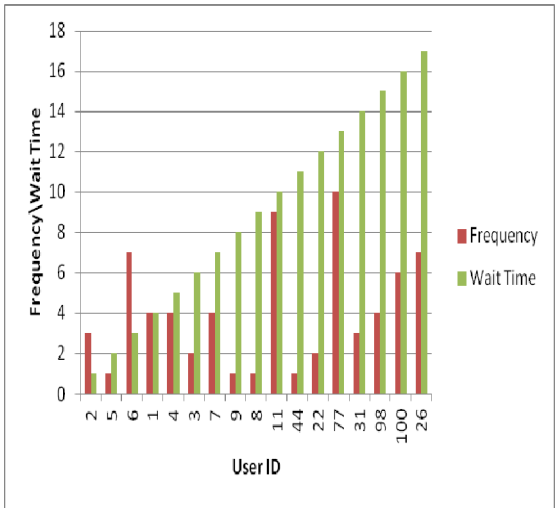


Figure 5

Conclusion

In this paper an algorithm is proposed which gives better services to the client who frequently makes request to resource broker for use of various services available in a cloud computing environment by different service providers. The average waiting of the client decreases and mutual trust is established between resource broker and client. Client will get timely and cost effective service and vendors' resource will be utilized efficiently.

References

[1] R. N. Calheiros, et al, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software: Practice and Experience 41.1 (2011), pp. 23-50.

[2] E. Afgan, "Role of the Resource Broker in the Grid." Proceedings of the 42nd annual Southeast regional conference. ACM, 2004

[3] Sotiriadis, S. Bessis, N. Antonpoulos, N., "Decentralized meta-brokers for inter-cloud: Modeling brokering coordinators for interoperable resource management," *Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th International Conference on* , vol., no., pp.2462,2468, 29-31 May 2012

[4] Nordin, B. I. Mohamad, A. B. Abdullah, M. I. Hassan , "Goal-based cloud broker for medical informatics application: a proposed goal-based request and selection strategy," *International Conference on Telecommunication Technology and Applications*. 2011.

[5] P. Hasselmeyer,"Removing the need for state dissemination in grid resource brokering," *Proceedings of the 5th international workshop on Middleware for grid computing: held at the ACM/IFIP/USENIX 8th International Middleware Conference*. ACM, 2007.

[6] K. M. Sim, "Agent-based cloud commerce," *Industrial Engineering and Engineering Management, 2009. IEEM 2009. IEEE International Conference on*. vol., no., pp.717,721, 8-11 Dec. 2009.

[7] S. G. Grivas, U. K. Tripathi, W. Holger, "Cloud broker: Bringing intelligence into the cloud," *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*. ol., no., pp.544,545, 5-10 July 2010

[8] S. K. Nair, et al. "Towards secure cloud bursting, brokerage and aggregation," Web Services (ECOWS), 2010 IEEE 8th European Conference on, vol., no., pp.189,196, 1-3 Dec. 2010

[9] H. K. Mehta, P. Kanungo, M. Chandwani, "Performance enhancement of scheduling algorithms in clusters and grids using improved dynamic load balancing techniques," *Proceedings of the 20th international conference companion on World wide web*. ACM, 2011

Cloud Based Self Driving Cars

[1]Naveen S Yeshodara, [2]Nikhitha Kishore
[1]Software Consultant, [2]Software Engineer
Data Center Management
Novell Software Development Center
Bangalore, India
[1]ynaveen@novell.com, [2]knikhitha@novell.com

[3]Namratha S N
Assistant Professor, Instrumentation Department
B M S College of Engineering
Bangalore, India
[3]namrathans@gmail.com

***Abstract*—This paper presents a novel idea for reducing the data storage problems in the self-driving cars. Self-driving cars is a technology that is observed by the modern word with most curiosity. However the vulnerability with the car is the growing data and the approach for handling such huge amount of data growth. This paper proposes a cloud based self-driving car which can optimize the data storage problems in such cars. The idea is to not store any data in the car, rather download everything from the cloud as per the need of the travel. This allows the car to not keep a huge amount of data and rely on a cloud infrastructure for the drive.**

***Index Terms*—Self-driving car, public cloud, security, privacy**

I. Introduction

The most coveted project of the recent times had been the Google's Self-Driving cars of course. The ability to travel in a more safe way, reducing the accidents and obeying the traffic rules, all without manual intervention is something quite amazing and that's what the self-driving cars do!!. Google's fleet of robotic Toyota Priuses has now logged more than 190,000 miles (about 300,000 kilometers), driving in city traffic, busy highways, and mountainous roads with only occasional human intervention [1]. The Google's vision of reduced car accidents, fuel consumption and congestion doesn't seem to be quite far.

When the idea of a self-driving car itself poses to be so amazing what is not much researched is where should the huge data that the car generates go and sit. According to the reports the car is expected to generate one gigabyte of data per second [2] which is really huge. Big-data is trying to come up with solutions [2] but we don't have them in a fully read form now. It is true that the car needs roadmaps, code and lot of other data to optimize it's 'without-human' driving.

This paper proposes a solution for this huge data generation problem in the self-driving cars. The idea is to use a cloud infrastructure to store everything that is needed for the car to take its long drive. There is no need to now worry about the huge data generation, the data purging or anything, there'll be a cloud to handle all that. The idea is to have the whole code in the cloud and download it based on the user's usability. The cloud can consist of anything that needs to get the car onto the road: Traffic rules for each region, Road map for auto drive of all locations of that state or country, live traffic data, signal information and all other data which is required by the car for self-driving.

When the driver starts the car the authentication is done and after this he can input the region to where the car has to take him and the exact location. The idea is that only the data required to take the car to the specific location has to be loaded to the car thus avoiding the car to carry a huge amount of data on the go. This idea can be widely used for any kind of update to CAR software, Traffic Rules and can be even used to analyze how much a car has travelled and its wear and tear information.

The whole is based on the simple assumption that there is a persistent network connectivity to the cloud and sufficient storage is available in the car to back the data if it is well known ahead that the car has to drive to a location with limited network availability. This paper is organized into the following sections; Section II will give the overview of the Google's self-driving car, Section III will detail the existing challenges in terms of data growth in the existing system, Section IV suggests a solution to the existing challenges in the current system and Section V will have the usability of this solution. Section VI will finally conclude the paper.

II. Self-Driving Car

This section will brief about how the Google's self-driving car works in roads where not every time things work fine. The very heart of the system is a laser range finder mounted on the roof of the car. This is what generated the 3D view of the road lying ahead the car. After generating the 3D map of its environment the self-driving car combines the view with the various world maps, generate different data models so that finally the car can run without hitting any obstacles, any humans crossing the road violating the traffic rules and driving observing the traffic rules in that country.

There are four sensors on the car: four radars, mounted on the front and rear bumpers, that allow the car to "see" far enough to be able to deal with fast traffic on freeways; a camera, positioned near the rear-view mirror, that detects

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

traffic lights; and a GPS, inertial measurement unit, and wheel encoder, that determine the vehicle's location and keep track of its movements [1].

The car not just uses the GPS technology alone to detect where it is positioned but also relies maps from different terrains so that the data is accurate. Just relying the GPS will not get the co-ordinates accurate [1]. Another significant point is that the car compares the data that it is collecting with the data that was pre-recorded so that the pedestrians can be differentiated from other obstacles like poles, mailboxes etc.

However the car will turn aggressive in situation where other vehicles are not going to reciprocate the same way how it obeys the traffic rules. The current solution for such real world situation is that the car will advance a bit to such disobedient cars to show what it really intends to do.

Figure 1 shows how the on-board computer in a self-driving car sees the pedestrians and other objects in its environment using the laser range finder.

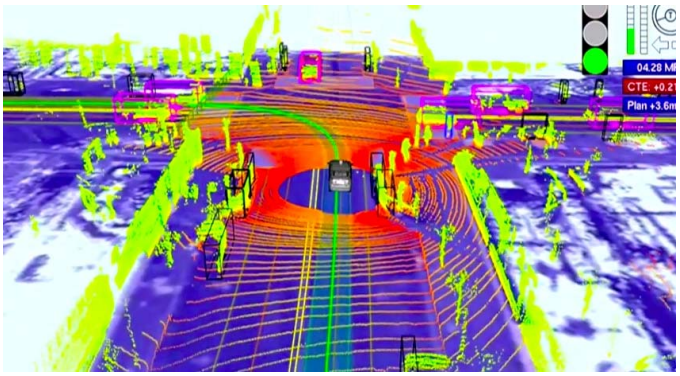


Figure 1: The image generated by the laser range finder in Google's self-driving car

However, the self-driving car cannot be just described as that of Google's. A lot of companies have predicted their own version of the autonomous cars in the new future. A list of official predictions are mentioned below [3]:

- Late 2014, Volvo will feature Adaptive Cruise Control with steer assist which will automatically follow the vehicle ahead in queues.
- Late 2014, The National Telecommunications and Information Administration is expected to set recommendations for setting aside broadband spectrum for autonomous cars.
- By 2015, Audi plans to market vehicles that can autonomously steer, accelerate and brake at lower speeds, such as in traffic jams.
- By 2015, Nissan expects to sell vehicles with autonomous steering, braking, lane guidance, throttle, gear shifting, and, as permitted by law, unoccupied self-parking after passengers exit.
- By Mid-2010's, Toyota plans to roll out near-autonomous vehicles dubbed Automated Highway Driving Assist with Lane Trace Control and Cooperative-adaptive Cruise Control.
- By 2016, Mobileye expects to release fully autonomous car technology.
- January 1, 2017 The National Highway Traffic Safety Administration hopes to mandate the adoption of Vehicle-to-Vehicle technology on all new automobiles.
- By 2017, Tesla plans an "autopilot" feature that handles 90% of miles driven. Tesla has partnered with Mobileye.
- By 2018, Google expects to release their autonomous car technology.
- By 2020, Volvo envisages having cars in which passengers would be immune from injuries.
- By 2020, GM, Mercedes-Benz, Audi, Nissan, BMW and Renault all expect to sell vehicles that can drive themselves at least part of the times.
- By 2025, Daimler and Ford expect autonomous vehicles on the market.
- 2035. IHS Automotive report says will be the year most self-driving vehicles will be operated completely independent from a human occupant's control.

The list underlines how much the idea of self-driving cars are gaining attention in the current scenarios. The autonomous cars are expected to reduce the traffic collisions, reduce the traffic congestion, eliminate the age-constraints for driving etc. Section III discuss in detail the challenges faced by the autonomous cars.

III. Potential Challenges For the Autonomous Cars

While there is a huge buzz going around the coming of self-driving cars the challenges that they pose cannot be ignored. Huge discussions are going around about the liability for damage, loss of privacy, cyber security etc. In this paper, we have focused on one of the challenges that is related to the data storage of the self-driving cars.

According to Mark van Rijmenam, founder of BigData-startups.com, self-driving cars will create 1 gigabyte of data per second. Multiply that by thousands, even millions, of cars, and the amount of data created daily comes to a staggering number [2].

When it comes to data storage this number is huge. To store this amount of data in the car on the drive will be an overhead. Google has already declared that while the car is on the track a set of complex algorithms will set on work to combine the various maps, GPS data and various other sources. Mobileye, which is a Dutch company that specializes in offering inexpensive cameras assisting self-driving cars, has witnessed a business raise to $400 million last year. So, this proves that the trend of self- driving cars is catching up fast and so in coming days we need to get ready to tackle the big data generated from these vehicles [4].

While Big-data is getting ready to provide some solid solutions to tackle this huge data storage problem, this paper proposes an idea which can optimize the amount of data that is used for driving and allow for much better autonomous driving.

IV. Live Automatic Self-Driving Car

The idea presented in this paper begins with the assumption that the network connectivity available should be 24/7. We have seen the potential advantages and disadvantages that the autonomous driving can introduce. Raw data is the essence behind the concept of a self-driving car. The data from various sources, GPS, maps of various terrains, laser range finder and sensors all pile up the data that is used by the software to turn to information to direct the car through real-life traffic.

In a Live automatic self-driving car, nothing is going to be stored in the car. So there is no fear of that car running and generating data and searching space to store it efficiently. SO where do the data go? Cloud is the solution that we propose. It is a known fact that the IT world is busy in expanding the cloud world to use just what is needed rather than worry about the infrastructure and hardware behind it. Same goes with the autonomous car also. From the perspective of a user, he need not worry about the data that is being dealt while the car is on a drive. The data storage issues should not bother a normal user who is unaware of such terminologies. In the idea that we propose here, every data is stored in the cloud. Whenever the data is required the software downloads it and make use of this. The set of complex algorithms, the pre-recorded GPS data and all that stuff are now going to be inside the cloud and the car downloads it whenever it requires. The only thing that is to be taken care of now is some external storage. This external storage is required because the car may need to locations where the network connectivity might fail. In such cases, the software downloads the area map and such details before the network fails so that it can work offline also.

The working of the proposed 'Live automatic self-driving car' is described below.

In the Google's self-driving car all the data resides inside the car. Switch on the car and the car takes the code, gets the maps ready, combine them with the live images and then start the drive. The only difference here is that all the data is pulled from the cloud, and yes that means the code also. When the driver starts the car, the on-board computer gets connected to the network, whichever is available, be it Wi-Fi, 4G or 3G. This starts the web console and asks for the authentication from the user. The user can sign in with the username and password registered with the cloud service provider. The user can then input in detail the location to where he needs to drive. The software then downloads the code to process the data, takes the pre-recorded data from the cloud that is just necessary to go to that locations, the world maps and start processing. Everything will be still in the control of user as how it is when the data is inside the car. A best advantage is that since nowadays V2V is gaining popularity the clouds can interact and make the self-driving safer.

The software will record the user's habits, where is goes often, where he stops, which are his favorite routes and all that and can use this data if the user wishes so to drive next time. The fact is that most users may need privacy and so the software needs to respect it. So when the driver is on a leisure trip such patterns can be put to use. The next important thing is sometimes the user may need to travel to a destination where the network connectivity is not available. In such cases the user can select a mode in which the software can download the data in advance and store in an external storage in the car. The software checks in intervals about the network connectivity and whenever it seems to have a problem it either asks the user to take up for a manual drive or downloads the data in advance to the external storage. Once the location is reached the software wipes off all the data from the car thus enhancing the security. But in case the driver wishes to keep the data the data will be encrypted so that even if the car is hacked also the data cannot be corrupted.

Figure 2 shows in detail how work flow of the Live automatic self-driving car.

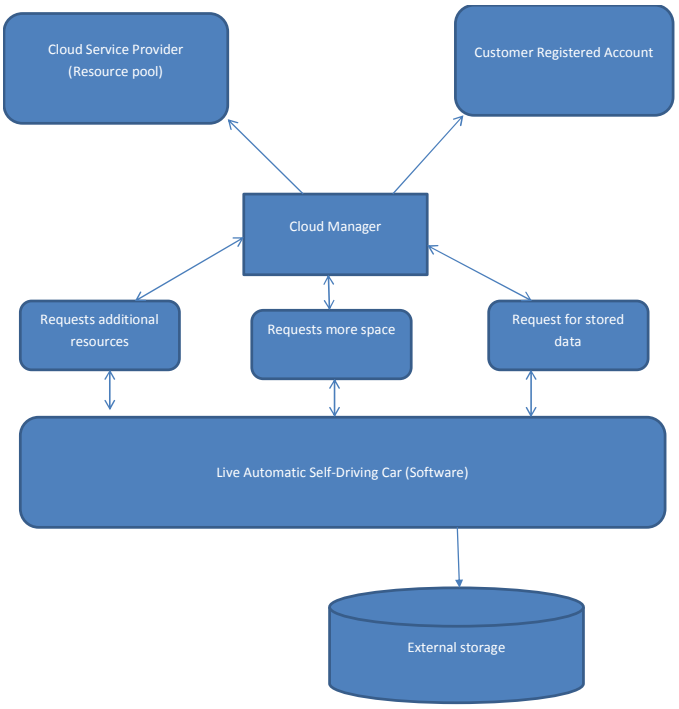


Fig. 1. Flowchart for a Live Automatic Self-driving car

A. The Web Console

When the user gets into the car and once he switch on the car, the web console appears in the on-board computer screen. It asks the user for the credentials with which he has registered with the cloud service provider. Once the authentication is done the web console should ask the user to input the destination to where he has to go. Once the user enters the destination, the web console should show options like whether the user is in an emergency or not. If the user is not in an emergency the web console pop-up will ask whether the user would prefer to use the pre-recorded behavioral patterns to be used while driving. This means that if the driver has stopped by an ice-cream parlor two-three times while going to the same

destination earlier the console would alert the user whether it should stop again in this drive a while before reaching the same ice-cream parlor.

The web console is an abstraction of what the self-driving car is all about. The user is not exposed to any of the technical details of how the car is going out in the street all alone. It is a friendly interface with which the user can interact on how to take the car to streets. The console also have other options like internet browsing, the much hyped vehicle to vehicle connectivity options, which not only makes the driving easier but allows the user to be connected with the world even while he is in the car.

Fig. 2. Data to be selected after authentication

B. The Network Selector

The users selections on the web console must now be send to the cloud and this requires a network connectivity. There are a lot of options for getting the car connected to the cloud, either the user can use tethering/hotspot, embedded, public wi-fi, 4G/3G services etc. Since a private Wi-Fi is still not available, the car will have to switch between public Wi-Fi or 3G/4G which is available at the time in a speed that the connectivity remains contiguous and the user need not interfere in the driving.

In such cases what is needed is a network selector. Cisco has already come up with such a client for V2V (Vehicle to Vehicle) solutions [4]. It claims that the Cisco on-board software solution can seamlessly switch between available 3G, 4G and other wireless networks based on cost and quality of service preferences. Such a client is what we call the network selector here. This will always ensure that the car is connected 24/7.

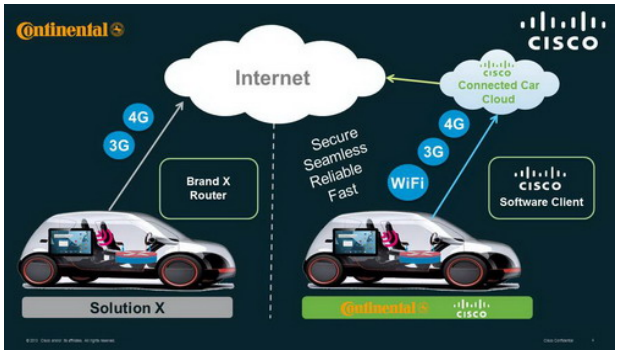


Fig. 3. Cisco software client to switch between available networks [9].

C. Authentication module

The authentication module is required for the car to authenticate the user of the car. This module is started once the car gets connected to the cloud as the user has to be registered with the cloud. The authentication module is not just a piece of software that remains local to the car, but after getting connected to the car it can authenticate other cars and send them to other locations also. This means that if the user wants to send his another car to a location to get is child from school or so, he can authenticate from his car and download the details to the other car also. But for this the requirement is that the other car has to be in the network at the same time.

D. External storage

It is quite normal that the user may decide to change the location to where he had decided to travel. In such cases there may be chances that the newly entered location has troubles with the network connection. This is processed using the self-driving car's on-board computer. In such cases a back-up is required. The software identifies the new location, send the details to the cloud manager and the cloud manager downloads the new data to the external storage. Once the network connection experiences any troubles the software does not fail, rather it uses the data that is now stored inside the car.

E. Encryption module

This module comes into play if the user wants the data to be stored in the car instead of wiping it off once the car is switched off. There may be cases when the driver has to go to the same destination the next day also. For example, the driver goes to office all five days in a week. He may think why to pull the data every day from the cloud. Rather keep the data in the car itself till Friday and wipe it off on Saturday. But the data has to be secured once it is inside the car and therefore it should be encrypted. This means that a hacker even if he gets the data also should not be able to make any use out of this.

The encryption is very important in the cases where the user is trying to drive another user profile. In this case both authentication and encryption has to go hand-in hand. Even if a wrong user can authenticate also the encryption/decryption

module has to be strong enough that both the cars can easily acknowledge each other without any misunderstanding.

When a car is self-driving then it is very important how it talks to other cars. There must be a mechanism in which one car has to tell that 'I am Car A and I want your authentication that you are Car B'. Once the authentication is done between both of them the communication should be encrypted. No hacker must be able to easily tap the communication and misguide the cars. For example, if car A tell car B that I will first turn and then you can take the turn and a hacker is misguiding the negotiation like, 'I am car A and you can take turn first', this will mean huge accidents are expected to happen on roads in the near future. A strict cryptographic policy is required for self-driving cars to move safely on the roads.

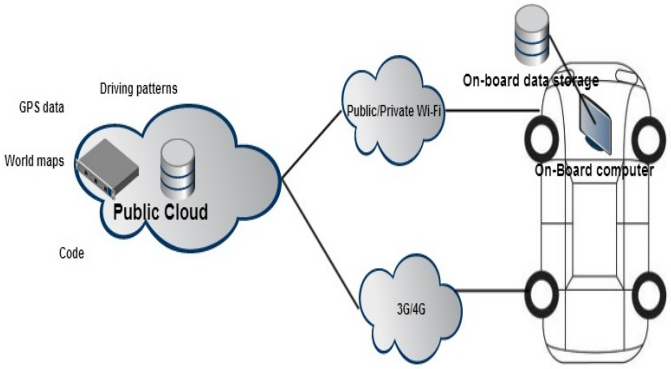


Fig. 4. Network Diagram for the proposed Self-driving car

V. Workflow For The Proposed Model

In this section, the workflow of the proposed cloud based self-driving car is explained.

1. The user gets into the car and switches on the car. Upon this step the login screen for authentication pops-up on the on-board computer screen.
2. The network connecter in the car will get connected to the internet using 3G/4G/Wi-Fi whichever is the best option available according to the user configurations.
3. Once the car is connected to the internet, the user will be given a number of profiles to select from. He can even login to the profile of his wife's car and send that car to a particular location.
4. The car connects to the cloud and authenticates the user details entered.
5. If the authentication is successful the screen to enter the location details are shown. The user enters the location/destination to where he has to reach.
6. The software pulls the details that are needed to drive to this location, like the code, the GPS data, the pre-recorded data, the user driving patters (if he wishes to), world maps etc to the storage in the car.
7. The car starts driving, the data is combined with the live map got from the laser-ranger finder for more accuracy.
8. In between if the user decides to change the location the software verifies whether the new location have data connectivity in a contiguous manner. Otherwise, the data for self-driving is downloaded to the car's external storage.
9. Once the car reaches a location, the software uploads all the data back to the cloud and get disconnected from the cloud.
10. If the user requires the data to stay in the car the data is encrypted and stored in the external storage.

This workflow is depicted as a diagram in Figure 5.

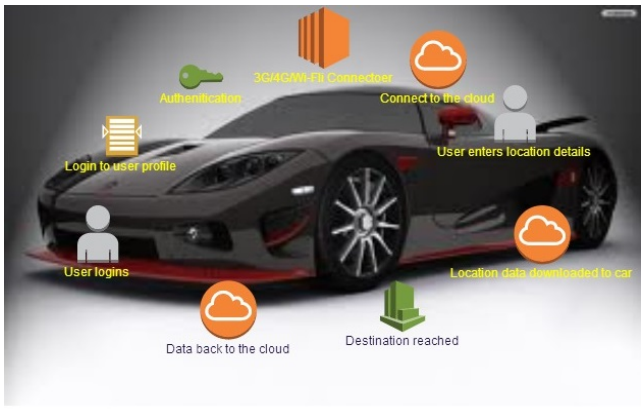


Fig. 5. Workflow Diagram for the proposed Self-driving car

VI. Connectivity Challenges

For the successful implementation of the proposed idea the network connectivity is a major challenge. The co-operation between the mobile network operators and automotive business is a key player here. However the structure between both the sectors are quite different for them to go hand-in-hand. The biggest gap will be the scope that is defined for both the business. The scope of business for an automotive company is worldwide whereas for a mobile network it is just limited to a particular country. For the same reason, a network that operates in India may not work in another region. So the car company cannot just make deal with a fixed mobile operator or so. Even in a country there may be local broadband network providers and some network provides who go everywhere within the country. Even the workflow for both the businesses are different. The development cycles are normally longer for the car companies and rapid for the network operators. A car may come with a 3G supporting software and the network operator may suddenly upgrade to the 4G support. Such rapid changes in the network industry has to be taken care when the self-driving cars are going to get connected with the cloud.

A very good option can be tethering. This means that the user just needs to have a smartphone with internet connectivity and the car can easily use this to get connected to the cloud. Since smartphones can easily switch between networks, it can be a good way to go. But can tethering help when the car is downloading huge amount of code, data, GPS records and all that? This is yet another challenge.

Another idea is the network connector that is now introduced by Cisco. Such a client will take the part of switching between the networks. Thus the changes in the networks can be easily handled and will be no more an issue. Any changes can be accommodated easily by upgrading the client whenever needed rather than changing the whole software provided with the car.

Since we also propose that a car when connected to the cloud can use the user profile of another car to drive it to another place the network connectivity is the heart beat of the proposed idea. The user at any time can authenticate the profile of another car once he know the details and can drive it to a particular location if the car is still in the network. One challenge here maybe how to get the car in the network. It is a huge waste if the user has to always put his car in the network.

VII. Usability and Challenges

In this section we present some use-cases for the proposed system and some challenges. Reducing the data storage trouble is the greatest advantage that the newly proposed system have over other solutions. Rather than optimizing the data generated the cloud-based system just moves the unused data out of the car and keeps it in the cloud. The data is downloaded only on-demand and therefore the amount of optimization done is huge. Some of the real-time use-cases for the proposed system is given below.

1. If driver wants to go to other location which is not selected during the startup, It should get connected to cloud and get the data downloaded some hours before he travels that way or it should inform him to switch to Manual mode.
2. This idea can be widely used for any kind of update to CAR software, Traffic Rules and even use to analyses how much a car has travelled and its wear and tear information
3. This idea can be extended to a level where, it drives to the nearby service center if it feels that there is some issue in the car or contact the nearest service station by itself and ask for help.
4. Also we can avoid traffic jams and even accidents if we know what's happening 1 KM or some distance ahead.
5. Once the car is stopped, the data is completely wiped out from car so that no one can steal or use the car without permission.
6. If driver wants to drive to same location on next day, he can mention that in application. Now the application will not wipe the data, rather will encrypt it and once driver authenticate it, the data will be decrypted.

Now we evaluate some of the potential challenges faced by the self-driving cars and see how the proposed solution can tackle them.

A. Cyber-security

One of the most potential threats to the autonomous cars is indeed the cyber-security. Most of the data will be inside the car and a hacker can tamper with the data to cause accidents. This a great threat as it can easy for terrorists or robbers to hack the data and cause huge mishaps.

According to the reports many security experts has already warned the security concerns that they have for the self-driving cars. According to most experts when there are self-driving cars that means they need a lot of negotiation between each other. What if this negotiation is not proper, instead of reducing road accidents, it will increase the number [5]. That means there needs to be a way to authenticate each other and convey who should move first or stop first etc. As long as security principles are not applied behind the

In the newly proposed system the data does not reside inside the database at all. Since the car downloads the data that is required only on the go and used up data is completely wiped off after the usage there is now no fear of mishandling of the data from outsiders. Even the data that is kept inside the external storage is also encrypted so that it is not easy for the hacker to quickly tamper the data.

The challenge while going for implementation with the public cloud is indeed the security concern. Even though the hacker does not have the data in the car, the whole data is now available in the cloud. The authentication can be easily broken with a smart level of expertise and thus can easily get the data. Not only that, your data is not now in your hand, it is available in the public cloud. This is a challenge for the proposed system.

Nowadays most of the enterprises are trying to move to the public cloud due to the level of agility that it provides. With this move, most of the public cloud service providers have increased their security levels. It might take much time for a hacker to hack into the entire system rather than trying to hack the data if stored inside the car. So in this way, storage of data in the car leads to vulnerability than when it resides in the cloud.

B. Loss of privacy

This is one factor that may be a curse in disguise of a boon. Most of the autonomous car ideas offer a feature that studies the users driving habits and take actions according to that. For example: if the user always stop by his favorite ice-cream shop when he travels to a particular location, the autonomous car has this feature to ask whether the user wants to stop by the ice-cream corner next time it goes through the same location. This may be pleasing sometimes, but not always. Suppose we late for a meeting and going in the car and the car starts asking you about your favorite restaurant, ice-cream shop etc it can be a

real nuisance. How to tackle the misuse of your private data is the biggest question.

In the newly proposed idea, all the data is residing in the cloud, including the users' private data. When going for an important meeting or any urgent situation the user can prevent the download of his private data so that the car straight away takes him to the location that waste time in analyzing the behavioral patterns of the user. This allows less data to be downloaded into the system saving more space. This portion of the data is now with the car, but also untouched which is even better.

C. Failures in network connectivity

The heart of the entire proposed system is in fact the network connectivity. If there is no network connectivity fails there should be ways in which such a situation can be handled peacefully. The switching between the networks need also to be very smoothly that it never affects the driving.

The car may be running in a public Wi-Fi connection and may enter a zone where Wi-Fi signals are not available. In such cases it is very necessary that the car immediately switched to a mobile network like 3G/4G. This hand-over needs to be taken very smoothly. Even a small second of failure in the road can cause a huge loss.

Imagine an autonomous car user deciding to go a place where he has not been before. In such cases if the new place does not have a persisting network connection and the user does not know how to drive it may be the ultimate challenge. In the newly proposed solution when the user enters the location details the software collects all the network details. Once it finds out that the area has problem with connectivity it downloads all the sufficient data to the external storage. In this case once the network fails also the car can drive without a driver with the code it had downloaded before.

As for the smoother hand-over between different networks the Cisco has introduced a new cloud client as discussed in the earlier section. In the proposed system we have included such a network selector than can smoothly conduct the transition between the networks so that the driving can be accurate and safe.

VIII. Conclusion

This paper proposed a solution for the data storage troubles that are encountered in a self-driving car. With many companies announcing dates for the official release of their self-driving cars, it has to be acknowledged as the technology of the future. Even though the challenges in building such a car is many the big question is where to store the huge amount of data that it generates. This paper suggests a cloud based self-driving car in which the data is completely taken away from the car. Everything is now stored in the cloud. The software talks with the cloud manager and gets the data that is required to go to a particular location. Only the details for that location is downloaded into the car. All the remaining data stays with the cloud. An external storage is also required in cases when the car has to travel to locations where the network connectivity is not contiguous.

References

[1] 'How Google's self-driving car works', *IEEE Spectrum blog*.

[2] 'Self-driving cars to create IGB of data per second', *FierceBigData,* Pam Baker, July 22, 2013

[3] http://en.wikipedia.org/wiki/Self-driving_car

[4] 'Connected car industry report 2013', *Telefonica.*

[5] Siva R.K Narla, 'The Evolution of Connected Vehicle Technology: From Smart Drivers to Smart Cars to… Self-Driving Cars', *ITE Journal,* July 2013.

[6] 'Connected cars: Business Model Innovation', *GSMA Connected Living Programme: mAutomotive.*

[7] 'Vehicle Information Exchange Needs for Mobility Applications: Version 2.0', U.S Department of transportation, Research and Innovative Technology administration.

[8] 'Exploring the connected car', *Cognizant 20-20 insights,* Nov 2012.

.

Cloud Partner Selection Algorithm for Dynamic Cloud Collaboration

Pramod Mane
Discipline of Computer Science and Engineering
Indian Institute of Technology Indore
Indore, India 453 441
Email: pramod@iiti.ac.in

Abhay Ratnaparkhi
CIO Innovation Lab
IBM
Pune, India 411 006
Email: abhay.ratnaparkhi@in.ibm.com

***Abstract*—The idea of composite cloud service has been emerging to reduce negative impact of cloud bursting. The novel idea of composite cloud services is achieved by forming a dynamic cloud collaboration platform among cloud providers. An important prerequisite of dynamic collaborative cloud formation is to reduce the cost of infrastructure and prevent loss to business enterprises owing to cloud bursting. The major concern in dynamic cloud collaboration is to minimize conflict among cloud providers and ensure each provider's benefit. In recent years several market based models have been proposed that deal with twofold objectives. First, conflict minimization among providers and Second, benefit maximization of the providers. However, existing combinatorial auction based market models that attempt to achieve dynamic cloud collaboration are computationally in efficient. In this paper we have proposed cloud partner matching algorithm to facilitate partner selection process. Our proposed cloud partner matching algorithm minimizes conflicts among cloud providers by mutual consent.**

I. Introduction

In recent years the concept of dynamic cloud collaboration is gaining importance among cloud providers to serve various combinations of demands of cloud consumers. There are several issues that adversely impact business. Firstly, Vendor-in- Lock oriented business model restricts cloud consumers from obtaining service on demand from other cloud providers. This restriction is disadvantageous not only to the consumer but also to the cloud provider. Secondly, cloud provider can be penalized on account of violation of contract (SLA) if the cloud bursts (a cloud-based infrastructure's inability to efficiently manage resource requirement). Thirdly, dynamic collaboration is gaining importance to cope up with unclear, chaotic and rapid changes in the cloud market environment along with fluctuating economic circumstances [1]. Hence, cloud providers are looking for a better business model that caters to the above mentioned issues. Dynamic collaboration among cloud providers provides a way for new corporate values that can address the rapidly fluctuating demands in cloud market. Dynamic collaboration leads to growth in business of cloud provider by reducing the cost of setup and maintenance on cloud infrastructure.

In cloud collaboration, each cloud provider shares its own local cloud service or resource with other cloud provider, and hence, it provides access to a much larger pool of resources that serves consumer requests on demand. Also, each participating cloud provider in collaboration can maximize its profit by offering its existing service capacities to the collaborative partner. There is also a provision in cloud collaboration in which cloud provider can satisfy a consumer demand by meshing up the existing services. Hence, cloud provider can efficiently tackle cloud bursting problem owing to availability of redundant clouds.

Changwoo Y. et al. [2] points out two major challenges. First is how to design an appropriate market model which facilitates dynamic collaborative platform. The second is to select suitable collaborative partners to provide service. Hassan M.M. et al. [3] proposed a novel CACM model (combinatorial auction cloud market model) that enables a dynamic cloud platform among cloud providers. In CACM model authors define single and group bid functions. Authors also talk about partner selection process that tries to fulfill various objectives. First, is to minimize total price, and second is to maximize collaborative performance. However, function takes order $O(nm)^3$ time to achieve objectives, where n is the number of cloud partners and m is the number of services.

However, the CACM model [3] neither considers mutual consent of partners in the process of partner selection nor discusses about profit distribution among group members. In real world there is a need of mutual consent of both partners to build tie-up or collaboration.

By incorporating mutual consent aspect in the process of partner selection, here we propose the mechanism to select the partner with mutual consent to minimize the conflict between cloud partners. The idea is to make use of the Gale-Shapley's 'deferred acceptance' algorithm [4] to select cloud partners. We have made a few changes in off-the-shelf algorithm to make it suitable for dynamic collaborative framework. The proposed algorithm takes polynomial time to find cloud partners.

II. Related Work

The idea of formation of market oriented dynamic collaborative platform is motivated from resource management in grid computing paradigm. Nepal S. et al. [5] proposed a protocol for agreement together with an efficient message piggy-backing algorithm to facilitate dynamic collaboration. In another work, Nepal S. along with John Z. [6] discussed the idea of localizing re-negotiation process. In the same work,

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

the SPIN model checker was utilized to implement CNA algorithm. In Grid computing context, Bubendorfer K. [7] proposed COARA (Collaborative Oversubscribing Resource Allocation Architecture), a market based resource reservation framework in the context of OpenGrid. COARA employs Vickery auction to provide way for combinatorial allocation of resources. In other work, Das A. et al. [8] proposed combinatorial auction based model for resource management in grid. In the proposed model, the user bids for each of the combinations of resources required for task completion. Changwoo Y. et al. [2] points out two issues. First how to find or design an appropriate market model for dynamic collaborative platform. Second, how to select suitable collaborative partners to provide service. Mohemmed H. et al. [3] proposed CACM model, based on combinatorial auction market model. CACM model defines method to select cloud partner for dynamic cloud collaboration. In the same work, authors highlight that the cloud partner selection in the context of dynamic cloud collaboration is different from other areas like virtual enterprise [9], manufacturing or supply chain [10] and the difference is owing to constitution of SLA.

III. Our Mechanism

A. Notation Summary

Notation	Summary
Time T	is divided and modeled as discrete time periods $T = \{t_1, t_2, ..., t_n\}$
$\mathcal{N}$	is the set of cloud providers indexed by $1, 2, ..., n$.
$P = \{P_1, P_2, ..., P_p\}$	is the set of primary cloud providers.
$N = \{N_1, N_2, ..., N_n\}$	is the set of non-primary cloud providers.
$S = \{S_1, S_2, ..., S_s\}$	is the set of services.
O_{N_i}	is the set of services that cloud provider i can provide at her end.
R_{P_i}	is the set of services that primary cloud provider $i \in P$ seek from any non-primary cloud provider $j \in N$.
Γ	is the set of service requirements that all primary cloud providers looking from non-primary cloud providers.
$i \succ_{k,s_i} j$	denotes that, for service s_i, user k more prefer to user i over j.
$i \prec_{k,s_i} j$	denotes that, for service s_i, user k less prefer to user i over j.

B. Model

Currently, some cloud provider(s) find business opportunity by a cloud broker. In CACM model [3] a cloud provider finds business opportunity by winning auction. The cloud provider, who found business opportunity is termed as primary cloud provider (the same term is followed in CACM model [3]). Cloud providers other than primary cloud providers are termed as non-primary cloud providers or secondary cloud provider. P is the set of primary cloud providers and N is the set of non-primary cloud providers. aWhereas $N, P \subset \mathcal{N}$ and $P \cap N = \phi$. When cloud consumers request multiple resources, primary cloud providers might not be in a position to serve these requested service requirements. In this case, primary cloud provider forms collaboration with other non-primary cloud providers. In collaboration non primary cloud provider acts as a partner cloud provider. P_i contains $\{< S_{P_i} >\}$ which represents that S_{P_i} is the set of services that P_i looks for from other non-primary cloud providers for cloud collaboration. On the other hand, N_i contains only $< O_{N_i} >$ and constitutes the set of services O_{N_i} that non-primary cloud provider own and she is interested to share these services with the primary cloud providers. Γ is a set of pairs $< m, s_i >$ such that $s_i \in S$ and $m : S \rightarrow \mathbb{Z}^+$. In other words, m counts how many time service $s_i \in S$ occurs.

In CACM model [3] system model for auction, represented by two actions. First, single and group bidding, and second, a partner selection. The partner selection issue discussed in CACM model does not take primary and non primary cloud provider's 'consent' into consideration for collaboration formation. However in real world there is need of consent of both members who are engaging in collaboration. We capture, the notion of 'consent' through choice theory and model it in the following way

- $N_i \succ_{P_i,S_i} N_j$ denotes that for service S_i, a primary cloud provider P_i prefers a non-primary cloud provider N_i over non-primary cloud provider N_j. In other words, when a primary cloud provider P_i searches for the service S_i from other non primary service providers and if there exist two non-primary cloud providers N_i and N_j, who have service S_i, primary cloud provider P_i prefers N_i for collaboration (to be partner) over N_j, The preference of P_i for partnership depends on collaboration cost, relation between N_i and N_j, etc.
- $N_i \prec_{P_i,S_i} N_j$ denotes that for primary cloud provider P_i who is searching for service S_i, secondary cloud provider N_i is less preferred than N_j.

On the other hand, like primary cloud provider, every non primary cloud provider N_i maintains preferences of primary cloud providers for sharing services owned by her O_{N_i}. In the case when, $P_i \succ_{N_i,S_i} P_j$ signifies that non primary cloud provider N_i for sharing her owned service S_i, prefers primary cloud provider P_i over cloud provider P_j and reason could be same as collaboration cost, relation between P_i and P_j, etc.

$P_i \prec_{N_i,S_i} Pj$ denotes that for service S_i, cloud provider P_i is less preferred than cloud provider P_j by non primary cloud provider N_i. The following matrices shows the tabular form where for each service cloud provider lists preferences over partner cloud provider. Note that primary cloud provider maintains choices for required services where as secondary or non-primary cloud provider maintains preference list for

those service which she would like to share with primary cloud providers.

$$P_i = \begin{array}{c} \\ S_i \\ . \\ . \\ . \\ S_n \end{array} \begin{array}{c} \begin{array}{cccc} 1^{st} & 2^{nd} & . & n^{th} \end{array} \\ \begin{pmatrix} N_1 & . & . & N_n \\ N_2 & . & . & N_j \\ N_7 & . & . & N_k \\ N_4 & . & . & N_l \\ N_1 & . & . & N_m \end{pmatrix} \end{array}$$

$$N_i = \begin{array}{c} \\ S_i \\ . \\ . \\ . \\ S_n \end{array} \begin{array}{c} \begin{array}{cccc} 1^{st} & 2^{nd} & . & n^{th} \end{array} \\ \begin{pmatrix} P_1 & . & . & P_n \\ P_3 & . & . & P_j \\ P_4 & . & . & P_k \\ P_5 & . & . & P_l \\ P_8 & . & . & P_m \end{pmatrix} \end{array}$$

Example 3.1:
Let us say, at prevailing situation there are five cloud providers present in market $\mathcal{N} = \{A, B, C, D, E\}$. Among them A, B are primary cloud providers and C, D, E are non primary cloud providers and these cloud providers are partitioned in two sets $P = A, B$ and $N = C, D, E$. A primary cloud provider A searching for services $\{S_2, S_3\}$ i.e. $A = \{< S_2, S_3 >\}$. Secondly, primary cloud provider B searching for services $\{S_2, S_3, S_4\}$ i.e.$B = \{< S_2, S_3, S_4 >\}$. On other hand, non-primary cloud provider C has $\{S_3, S_2\}$ services and is ready to share these services with a primary cloud provider i.e. $C = < S_2, S_3 >$. Non primary cloud provider D has $\{S_3, S_4\}$ resource to share with primary cloud provider i.e. $D = \{< S_3, S_4 >\}$. A non primary cloud provider E looks for a primary cloud provider with whom she would like to share resource $E = \{< S_2, S_4 >\}$. Hence $\Gamma = \{2.S_2, 2.S_3, 1.S_4\}$ is the set of service requirements that all primary cloud providers are looking for from non-primary cloud providers
Here service $S_2 \in \Gamma$ is a requirement of A and B where as C and E are potential partners who have the service S_2. The service $S_3 \in \Gamma$ is a requirement of A and B where as non primary cloud provider C and D can provide the service. The service S_4 is a requirement of only B, where as D and E are potential service providers of it.

C. Cloud Partner Matching Algorithm

In this section we introduce the cloud partner matching algorithm. We make use of the Gale-Shapley's 'deffered matching' algorithm to develop procedure for partner selection in dynamic collaboration. The Gale-Shapley's 'deffered matching' algorithm is proposed in the context of college admission and stability of marriage. In cloud partner matching procedure, a primary cloud provider(s) initiates process of dynamic cloud collaboration. A primary cloud provider asks secondary cloud provider for collaboration where as secondary cloud provider accepts or rejects the proposal. A primary cloud provider follows the same procedure for every resource that he looks for from non primary cloud provider. Algorithm 1 describes process of partner matching and evolution of dynamic cloud collaboration.

Algorithm 1 CPMP (Cloud Partner Matching Process)

```
p ∈ P and n ∈ N
while Γ is not empty do
  assign each primary cloud provider and secondary cloud
  provider to be free
  while some primary cloud provider p is free do
    non-primary cloud provider,n := first non primary
    cloud provider on p's list to whom p has not asked
    for collaboration
    if non primary cloud provider n is free then
      n and p collaborate
    else
      if n prefers p over her collaborator p' then
        p and n tie-up in collaboration and p' becomes
        free
      else
        n dismisses p and (p becomes free)
      end if
    end if
  end while
end while
```

D. Case Study

If we consider Example 3.1 scenario,
$A = < S_2, S_3 >$ and $B = < S_2, S_3, S_4 >$
Where as
$C = < S_2, S_3 >$, $D = < S_3, S_4 >$, $E = < S_2, S_4 >$
and $\Gamma = \{2.S_2, 2.S_3, 2.S_4\}$
Following are the preference matrix of primary cloud providers A and B respectively:

$$A = \begin{array}{c} \\ S_2 \\ S_3 \end{array} \begin{array}{c} \begin{array}{cc} 1^{st} & 2^{nd} \end{array} \\ \begin{pmatrix} C & E \\ C & D \end{pmatrix} \end{array} B = \begin{array}{c} \\ S_2 \\ S_3 \\ S_4 \end{array} \begin{array}{c} \begin{array}{cc} 1^{st} & 2^{nd} \end{array} \\ \begin{pmatrix} C & E \\ D & C \\ E & D \end{pmatrix} \end{array}$$

Where as the preference matrix of secondary cloud providers C and D and E respectively are as follows:

$$C = \begin{array}{c} \\ S_2 \\ S_3 \end{array} \begin{array}{c} \begin{array}{cc} 1^{st} & 2^{nd} \end{array} \\ \begin{pmatrix} B & A \\ B & A \end{pmatrix} \end{array} D = \begin{array}{c} \\ S_3 \\ S_4 \end{array} \begin{array}{c} \begin{array}{cc} 1^{st} & 2^{nd} \end{array} \\ \begin{pmatrix} B & A \\ A & B \end{pmatrix} \end{array} E = \begin{array}{c} \\ S_2 \\ S_4 \end{array} \begin{array}{c} \begin{array}{cc} 1^{st} & 2^{nd} \end{array} \\ \begin{pmatrix} A & B \\ B & A \end{pmatrix} \end{array}$$

1) Collaboration for the example: When we apply cloud partner matching algorithm in following manner
Matching for Service S_2
Step 1: Cloud provider A asks her first preference C. If C is free, A is assigned to C.
Step 2: Cloud provider B asks her first choice C. If C prefers B over A, then C accepts partnership proposal of B, and A becomes free.
Step 3: A is now free, then she asks her second choice E for partnership, E is free and assign to A.

Step 4: As there is no free primary cloud provider remained, inner while loop is over.
Step 5: At the end of inner while loop, for service S_2, $\{BC\}$ and $\{AE\}$ partnership occurs.
We follow the same procedure then we find collaboration $\{AC\}$ and $\{BD\}$for service S_3 where as for service S_4 we obtain tie-up $\{BE\}$. The grand coalition for primary cloud provider A is $\{ACE\}$ for all services that A looks for. And the grand coalition for primary cloud provider B is $\{BCDE\}$ for all services that B looks for.

E. Conflict Concept

Here we present our view about the notion of 'Conflict'. We assume that in current cloud collaboration, conflict occurs between cloud providers for service S_i, if P_i and N_i prefer each other over their current cloud partner, for $i = 1, 2$.
In other words, let us say for service S_i the current collaboration is done in following way: $\{P_1, N_2\}$ and $\{P_2, N_1\}$. Now at one side, if P_1 prefers N_1 and N_1 also prefers P_1. And on the other side, if P_2 prefers N_2 and N_2 prefers P_2. Then we can say that pairs $\{P_1, N_2\}$ and $\{P_2, N_1\}$ are in conflict and each P_i and N_i are with conflict collaborators(partners).

IV. Analysis and Discussion

Here we claim that proposed cloud partner matching algorithm minimizes conflict and takes $(n^2)^{|\Gamma|}$ time to find cloud partner for cloud collaboration.

Proposition 4.1: CPMP method minimizes conflict in collaboration.

Proof Suppose P_1 and N_1 are with conflict collaborators for service S_i.

- Case-1: P_1 asks N_1 for collaboration. N_1 rejects proposal of P_1 for collaboration (right away or later in the process), this implies that N_1 prefers her current cloud collaborator to P_1, this implies that N_1 is not with conflict collaborator.
- Case-2: P_1 never asks N_1 for collaboration. The above case signifies that P_1 prefer her cloud collaborator to N_1 which implies that P_1 is not with conflict collaborator.
- In either case, both P_1 and N_1 are with conflict collaborator which is a contradiction.

Proposition 4.2: CPMP algorithm takes $(n^2)^{|\Gamma|}$ time to find collaboration.

Proof Assume that there are n primary cloud providers and n secondary cloud providers for every service S_i then every primary cloud provider take at most $n-1$ steps to find her partner for collaboration then algorithm take $(n(n-1))$ steps to find single partner for every primary cloud provider. However the process of finding partner or collaborator continues for every service $S_i \in |\Gamma|$, hence the algorithm takes $O(n^2)^{|\Gamma|}$ time to find collaboration.

V. Conclusion

The main goal of the paper is to minimize conflict between cloud partners in partner selection process. In the current study we incorporate the notion of preferences of cloud partners to present the concept of mutual consent. We show that cloud partner matching algorithm minimizes conflict between cloud partners while forming cloud collaboration. The study has gone some way towards enhancing our knowledge of partner selection process and the role of preference or mutual consent between partners in dynamic cloud collaboration. Finally, a number of important limitations need to be considered. First, the proposed algorithm is based on two assumptions that number of primary and secondary cloud providers are equal, which is ideal condition. It also assumes that for a service each primary cloud provider has an ordered preference over secondary cloud providers. For unequal set of partners there could be unmatched primary or secondary providers. This indicates that the request can not be fulfilled if we would like to have a stable matching. There could be incomplete preferences (some cloud provider do not wish to collaborate with some other providers) or ties (no strict preference over other). The algorithm can be easily extended to take care of such cases.

Acknowledgment

We would like to offer our special thanks to Ms. Shruti Bhilare, Department of Computer Science and Engineering, IIT Indore, India. for her valuable input.

References

[1] Y. Yamazaki, "Dynamic collaboration: the model of new business that quickly responds to changes in the market through the integrated it/network solutions provided by nec." *NEC journal of advanced technology*, vol. 1, no. 1, pp. 9–16, 2004.

[2] C. Yoon, M. M. Hassan, H. Lee, W. Ryu, and E.-N. Huh, "Dynamic collaborative cloud service platform: opportunities and challenges," *ETRI journal*, vol. 32, no. 4, pp. 634–637, 2010.

[3] B. S. Hassan, Mohammad Mehedi and E.-N. Huh, "A market-oriented dynamic collaborative cloud services platform," *annals of telecommunications-annales des telecommunications*, vol. 65, no. 11-12, pp. 669–688, 2010.

[4] D. Gale and L. S. Shapley, "College admissions and the stability of marriage." *American Mathematical Monthly, JSTOR*, vol. 120, no. 5, pp. 386–391, 2013.

[5] S. Nepal, Z. John, and C. Jonathan, "A distributed approach for negotiating resource contributions in dynamic collaboration," in *Eighth International Conference on Parallel and Distributed Computing, Applications and Technologies*, 2007, pp. 82–86.

[6] S. Nepal and Z. John, "A conflict neighbouring negotiation algorithm for resource services in dynamic collaborations," in *IEEE International Conference on. Services Computing 2008. SCC'08.*, vol. 2. IEEE, 2008.

[7] K. Bubendorfer, "Fine grained resource reservation in open grid economies," in *Second IEEE International Conference on e-Science'06.* IEEE, 2006, pp. 81–81.

[8] A. Das and G. Daniel, "Combinatorial auction-based protocols for resource allocation in grids." in *19th IEEE International Parallel and Distributed Processing Symposium.* IEEE, 2005, pp. 8–pp.

[9] J. B. Davies, "A multi-objective optimization model of the partner selection problem in a virtual enterprise and its solution with genetic algorithms (by zhao fuqing; hong yi; yu dongmei)," *The International Journal of Advanced Manufacturing Technology*, vol. 37, no. 11, pp. 1220–1220, 2008.

[10] R.-C. W. Chang, Sheng-Lin and S.-Y. Wang, "Applying fuzzy linguistic quantifier to select supply chain partners at different phases of product life cycle," International Journal of Production Economics, vol. 100, no. 2, pp. 348–359, 2006.

Cloudifying Apps - A study of design and architectural considerations for developing cloud enabled applications with case study

Narsimha Reddy Challa , Venkatesh Nuthula
IBM India Limited
Hyderabad
Narsimha.Reddy@in.ibm.com
nuvenkat@in.ibm.com

***Abstract*—With the emergence of Cloud Computing technology many enterprises started moving their applications to the cloud to gain the benefits of hosting applications online versus having to have physical hardware or build out infrastructure. This new technology trend presents new challenges to application developers to enable applications in the cloud. Building applications for the cloud requires a major paradigm shift and new thinking about the application design, system architecture and needs an emphasis on leveraging massive scale. Building scalable applications for the cloud requires solid engineering and design by addressing the Statelessness, Redundancy, Resiliency, Server failures, New database approach, security, fast-changing platforms and dealing with different frameworks. While cloud deployments can abstract developers from having to deal with infrastructure issues, developers can focus on innovation and business logic instead of worrying about plumbing and infrastructure such as the operating systems, hardware etc. This paper is targeted towards *cloud application developers and architects* who are responsible for developing brand new cloud applications as well as migrating existing applications to clouds. The focus of this paper is to highlight design principles and best practices applicable to application development in cloud environment.**

Keywords—Application, Cloud, Compute, Database, Design, Elastic, File System, IaaS, Load Balancing, PaaS, Resilient, SaaS, Scalability, Security, Stateless, Storage, VM, Virtual Machine Image, Virtualization

I. Introduction

Cloud computing, which involves accessing applications and services in Cloud Data Centers over the Internet, certainly has become the hot technical trend in the industry. A study has revealed that Software as a Service (SaaS) and cloud-based business application services will grow from $13.4 billion in 2011 to $32.2 billion in 2016, a five-year CAGR of 19.1% [10]. Due to a large growth rate expected in the volume of cloud applications in next few years, it brings in both new benefits and responsibilities for developers and architects. This paper is targeted towards cloud application developers and architects who involved in developing or transforming enterprise class applications to cloud environments. This paper presents the design considerations inherent in designing cloud applications in order to take advantage of the strengths of cloud computing by highlighting best practices and design considerations in creating new cloud applications or migrating existing applications to the cloud. And, discusses the benefits of following such practices in terms of scalability, resilience, security and economics.

Throughout this paper various design and architectural considerations using Open Cloud Foundry Services, Amazon Web Services, Microsoft's Azure Cloud, Google's Cloud Services and IBM's PaaS offering Bluemix will be discussed.

II. Existing Issues / Challenges

Many enterprises are moving their applications to cloud to gain the benefits of hosting applications online versus having to have physical hardware. However, moving to the cloud can present significant challenges like scalability, performance and security in multi tenancy model that can impact productivity across their business if not designed and developed properly.

III. Proposed Design considerations for Cloud Applications

There are various challenging design aspects to be addressed in the designing a cloud application or porting an existing application to cloud environment.

A. Use Standard Application Frameworks

Generally, applications written in supported frameworks often run unmodified on Cloud environments, if the application design follows a few simple guidelines. Following standard frameworks makes an application cloud-friendly, and facilitates deployment to cloud environments. Generally, the standard frameworks will abstract most of the plumbing concerns related to infrastructure. Applications typically depend on services such as databases or third-party SaaS providers. When a developer provisions and binds a service to an application, the service broker for that service is responsible for providing the service instance irrespective of underlying infrastructure.

1) Software as a Service (SaaS): After an application has been identified as a candidate for cloud enablement, and if SaaS based alternatives exist, it is worth considering whether the SaaS alternatives can meet both business and technical needs. Such a change is no longer an application migration

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

but, more of a replacement of the existing application with a SaaS option. There might still be a need to migrate existing data to the new application. SaaS removes the need to manage both the application and the infrastructure on which the application is deployed. This approach can be attractive, but certain criteria, such as service-level agreements (SLAs), data portability, and long-term costs, must be carefully evaluated when considering an SaaS deployment.

2) Platform as a Service (PaaS): Platform as a Service (PaaS) is an application development and deployment platform delivered as a service to developers over the Web. It facilitates development and deployment of applications without the cost and complexity of buying and managing the underlying infrastructure, providing all of the facilities required to support the complete life cycle of building and delivering web applications and services entirely available from the Internet. This platform consists of infrastructure software, and typically includes a database, middleware and development tools. In this model, the service provider manages the application platform software and might provide access to common application services.

Case Study: Cloud Foundry – openPaaS

Cloud Foundry is the industry's first open Platform as a Service (openPaaS) to deploy and scale applications in seconds, without locking yourself into a single cloud. Providing a choice of developer frameworks, application services and deployment clouds. Cloud Foundry simplifies application development and makes it faster and easier to build, test, deploy and scale applications. Cloud Foundry already supports multiple application services MySQL, PostgreSQL , MongoDB, Redis and RabbitMQ that offer the database and messaging capabilities. Developers can easily bind their applications to one of these services during the deployment.

The novelty of Cloud Foundry is that it provides choice for Developers and organizations like

Development Frameworks - Cloud Foundry supports Java™ code, Spring, Ruby, Node.js, and custom frameworks.

Application Services - Cloud Foundry offers support for MySQL, MongoDB, PostgreSQL,Redis, RabbitMQ, and custom services.

Clouds - Developers and organizations can choose to run Cloud Foundry in Public, Private, VMWare and OpenStack-based clouds.

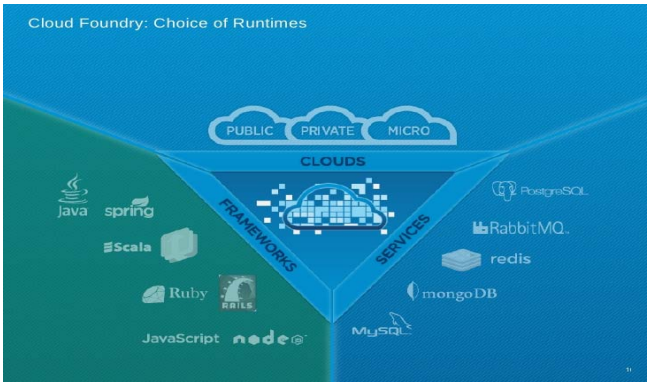


Fig. 1 . Cloud Foundry Runtime Choices

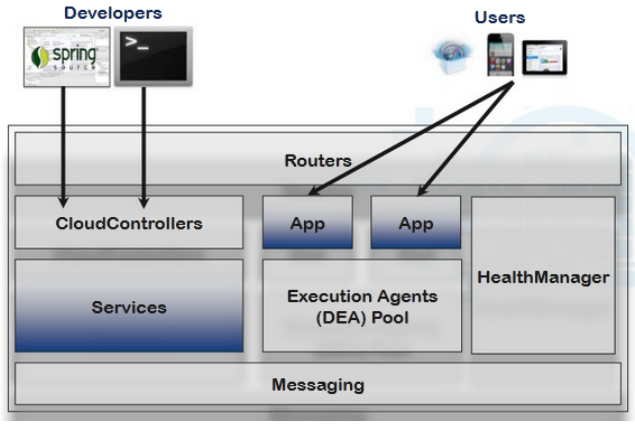


Fig. 2 . Architecture of Cloud Foundry

B. Use Virtualized Infrastructure

By using Virtualized Infrastructure an application is abstracted from underlying hardware, operating system, storage, and network. This will enable flexibility in deployment which allows business services to move dynamically across virtualized infrastructure in an efficient manner, based upon predefined policies that ensure specific service level objectives are met for these business services.

By using CloudFoundry PaaS services in developing Cloud Applications, there is no binding to any underlying physical/virtual hardware. The underlying Cloud can be OpenStack, VmWare, AWS, RackSpace or any other cloud that supports CloudFoundry services.

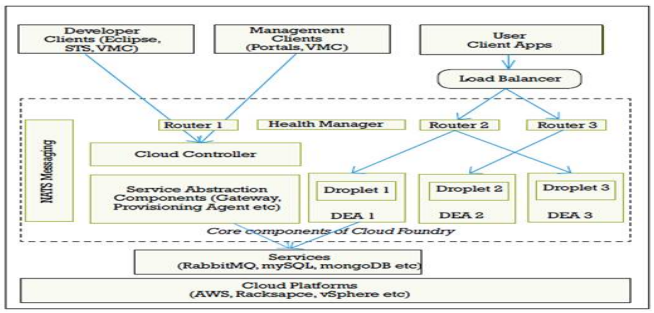


Fig. 3 . Cloud Foundry Layers

C. Design for Statelessness

A cloud application needs to be stateless. The reason being, stateful applications poses problems in case of failures and can't be restarted in a seamless manner. Basically, there is no concept of a local storage, registry etc. for cloud based applications. But that's all encapsulated by being a stateless application. In a stateless application, there is no dependency between requests. Failed request may need to be started on a different node. This model of application architecture, where storage is abstracted from the application, is referred to as statelessness and it's integral to the value proposition of designing applications for the cloud.

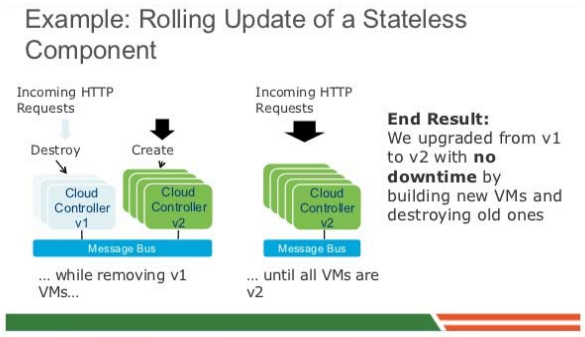


Fig. 4. Rolling Update of a Stateless Component

For example, Cloud Foundry does not persist or replicate HTTP session data. If an instance of an application crashes or is stopped, any data stored for HTTP sessions that were sticky to that instance are lost. When a user session that was sticky to a crashed or stopped instance makes another HTTP request, the request is routed to another instance of the application.

However, some applications always maintain some kind of state, which is why it requires databases or object stores to store some state information for maintaining user information etc. But, parts of the applications that needs to scale, such as the Web front end, are stateless in the cloud and can be considered for scaling.

D. Avoid Local File Storage

Generally, the concept of storage is not same in the cloud. Local file system storage is short-lived. When an application instance crashes or stops, the resources assigned to that instance are reclaimed by the platform including any local disk changes made since the application started. When the instance is restarted, the application will start with a new disk image. Although your application can write local files while it is running, the files will disappear after the application restarts.

The basic difference between local and cloud applications are, storage and networking resources are abstracted in a cloud. So, applications should not tie into any specific piece of physical hardware in clouds. Basically, this requires an abstraction to disassociate storage from the application itself. Instead, storage is spread across multiple cloud nodes, such that the failure of any single node has negligible impact on the whole.

Case Study: : Cloud Foundry – openPaaS

Applications running on Cloud Foundry should avoid writing files to the local file system. And, instances of the same application do not share a local file system as they could be running in different DEA nodes, in different virtual machines, and on different physical machines.

An alternative to user local file system is to use a Cloud Foundry services such as the MongoDB document database or a supported relational database (MySQL or vFabric Postgres). If your application needs to communicate across different instances of itself (for example to share state etc), consider the use of a cache like Redis or make use a messaging-based architecture with RabbitMQ.

E. Decouple Application Components

Another architectural consideration is, parts of an application (like web front end, application server etc.) might be in many places in the cloud. For example, a presentation layer might be on Facebook, storage could be on Amazon.com's S3, and application logic could run somewhere else entirely. In such decoupled architecture, various components or various instances of the same application needs to communicate.

1) Cloud Messaging using Cloud Foundry with RabbitMQ:

If the application needs to communicate across different instances of itself (for example to share state), consider a cache like Redis or a messaging-based architecture with RabbitMQ.

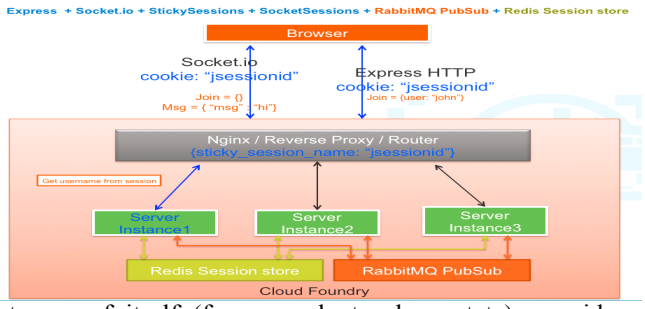


Fig. 5. Application architecture with RabbitMQ Messaging

F. Design for High Availability (HA) and DR

The other challenging aspect to be addressed in designing a cloud applications is redundancy as commodity servers can be deployed in cloud data centers. It is always possible that these servers can go down, so it is important to design cloud applications to have enough redundancy. This can be addressed by running redundant instances of an application to increase it's availability. Also, to avoid the risk of an application being unavailable during upgrade processes, more than one instance of an application needs to be run. The advantage of this approach is, if an instance in the cloud fails, an application that's designed for the cloud simply fails over to another instance without any kind of user intervention. Another advantage is, it is possible to replicate the environment in other location within minutes by taking advantage of geographic distribution with a lower cost option for maintaining a fleet of DR servers and data storage.

1) HA with CloudFoundry MongoDB service: Using CloudFoundry MongoDB service, High Availability can be achieved through Sharding and ReplicaSets. New instances can be deployed on Demand.

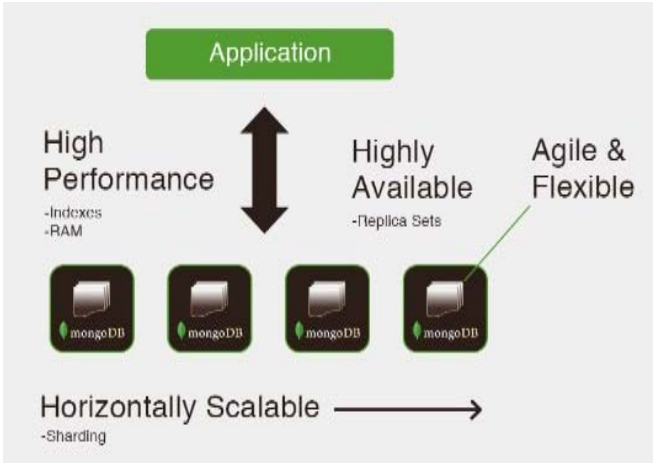


Fig. 6. CloundFoundry MongDB Service

MongoDB Features

Using the concept of sharding, MongoDB divides the data set and distributes the data over multiple servers, or shards. Each shard is an independent database, and collectively, the shards make up a single logical database. The benefits of sharding are

- Automatic balancing for changes in load and data distribution
- Easy addition of new nodes and scaling out to thousands of nodes.
- No Single points of failure and provides automatic failure.

ReplicaSets provide an automated method for storing multiple copies if user data. A ReplicaSet consists of two or more nodes that are copies of each other. Using Replication distributed Read Load/Read Scale achieved. Replica sets provide Disaster Recovery.

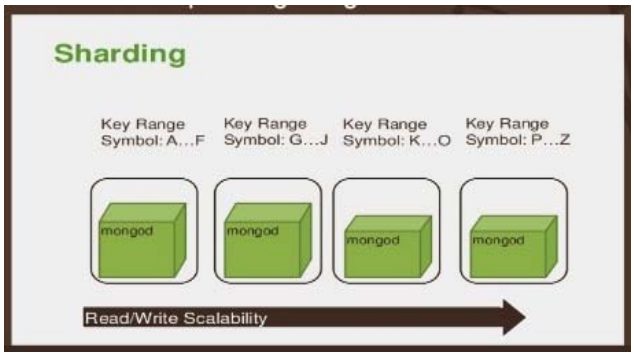


Fig. 7. MongoDB Sharding Feature

G. Avoid Relational Databases

In clouds, there can be multiple instances of the same application running but, do not share a local file system. Each application instance runs in it's own isolated environment like a separate VM or container. Each application instance has it's own local file system and when a file modified by one instance is not seen to other instances of the same application. As long as the files are temporary, this should not be a problem. However, if the application needs to persist data in the files across application restarts, or the data needs to be shared across all running instances of the application, the local file system or relational databases does not solve the problem. Under highly loaded conditions, relational databases (Ex: MySQL) perform poorly due to one-row-at-a-time approach.

1)CloudFoundry's MongoDB Service: Instead,a distributed database such as MongoDB document database using Cloud Foundry service allows data written simultaneously to different nodes. This will increase write speed without the problem of one-row-at-a-time approach. The following diagram shows the response times for Sql and MongoDB [14]

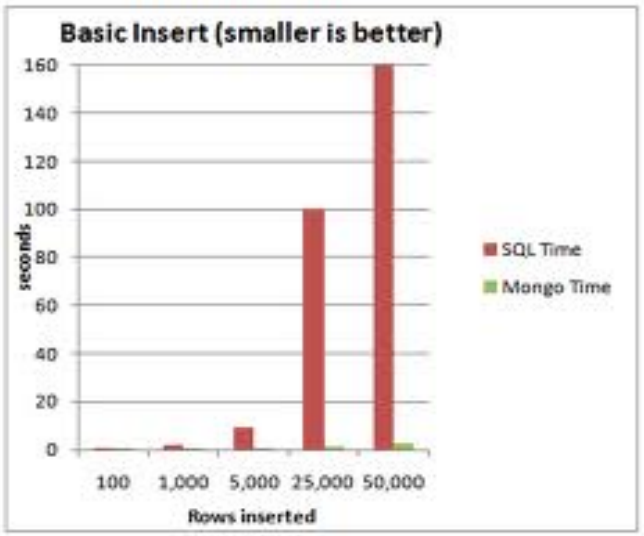


Fig. 8. Response Times for SQL and MongoDB

2) Amazon S3 Distributed Data Store: Amazon S3 is highly durable and distributed data store. With a simple web services interface, you can store and retrieve large amounts of data as objects in buckets (containers) at any time, from anywhere on the web using standard HTTP verbs. Copies of objects can be distributed and cached at 14 edge locations around the world by creating a distribution using Amazon CloudFront service. It gives any developer access to the same highly scalable, reliable, secure, fast, inexpensive infrastructure that Amazon uses to run its own global network of web sites.

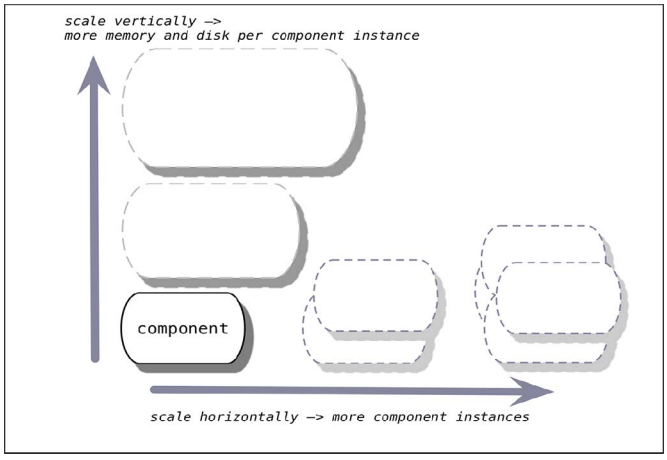
3) Google Cloud Storage: Google Cloud Storage, Dropbox, or Box can be used instead of relational databases. Another point is that, Google App Engine abstracts away not only the actual physical hardware but any concept of a machine. That means a developer uploads code, then Google manages it and splits up the database. With App Engine, developers uses Google's Big Table data store for persistent storage.

4) Azure Storage Engine: Another example, Azure presents developers with a different perspective on databases than the standard relational model. The Azure storage engine does not

use a standard relational database, so, a lot of the stuff you would do if you were developing a standard application using a standard relational database just doesn't make a lot of sense anymore. With Azure there's no guarantee that the data that you're going for lives in any particular location or data center or any particular device. The Azure storage engine is different than the SQL Data Services cloud-based version of SQL Server planned by Microsoft, so developers need to be careful as to which they're writing for. As an example, Azure stores a 1MB file as a blob, unlike SQL Server, which stores it in a table.

H. Design for Scalability

Conceptually, the cloud is designed to scale to any extent. However, it may not be possible to leverage all that scalability in infrastructure alone unless the application architecture is scalable. As a cloud application architect, identify bottlenecks in application architecture by identifying the components in which it is not possible to leverage the on-demand provisioning capabilities. Refactor the application in order to leverage the scalable infrastructure and take advantage of the cloud [6]. As an architect, design your application to scale up and down to match your unexpected demand without any human intervention. Auto-scaling encourages automation and drives more efficiency to address unpredictable traffic patterns, and the demand for faster response times.



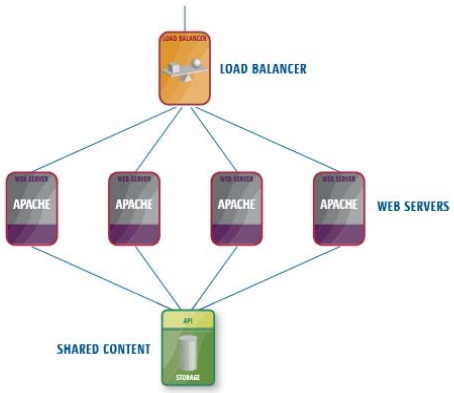


Figure 9. A very common use of parallelization and load balancing is horizontally scaled Web servers.

1) Scaling platform capacity with CloudFoundry:

You can scale platform capacity vertically by adding memory and disk, or horizontally by adding more VMs running instances of Cloud Foundry components. To scale the Cloud Foundry platform for high availability, the actions you take fall into three categories.

- For components that support multiple instances, increase the number of instances to achieve redundancy.
- For components that do not support multiple instances, choose a strategy for dealing with events that degrade availability.
- For database services, plan for and configure backup and restore where possible. For database services deployed outside Cloud Foundry, plan to leverage your infrastructure's high availability features and to configure backup and restore where possible.

I. Security

In a multi-tenant environment, cloud architects often express concerns about security. Security should be implemented in every layer of the cloud application architecture. Physical security is typically handled by your service provider, which is an additional benefit of using the cloud. Network and application-level security is your responsibility and you should implement the best practices as applicable to your business.

If you need to exchange sensitive or confidential information between a browser and a web server, configure SSL on your server instance. You'll need a certificate from an external certification authority like VeriSign or Entrust. The public key included in the certificate authenticates your server to the browser and serves as the basis for creating the shared session key used to encrypt the data in both directions. If you are concerned about storing sensitive and confidential data in the cloud, you should encrypt the data (individual files) before uploading it to the cloud.

For example, using AWS create a Virtual Private Cloud by making a few command line calls (using Amazon VPC). This will enable you to use your own logically isolated resources within the AWS cloud, and then connect those resources directly to your own datacenter using industry-standard encrypted IPSec VPN connections. You can also setup an OpenVPN server on an Amazon EC2 instance and install the OpenVPN client on all user PCs.

For example, encrypt the data using any open source30 or commercial PGP-based tools before storing it as Amazon S3 objects and decrypt it after download. This is often a good practice when building HIPPA-Compliant applications that need to store Protected Health Information (PHI).

J. Elasticity

Elasticity is one of the unprecedented concepts that have emerged due to the dynamic nature of the cloud and became

one of the fundamental properties of the cloud. Using the elastic properties of underlying cloud, an application can be made elastic by bundling the operating system, application software and associated configuration settings into virtual appliances and provisioning additional instances as well as decommission them on demand. Virtual Instances can be launched in one or more geographical regions. If you cannot embrace the change and implement elasticity in your application architecture, you might not be able to take the full advantage of the cloud. As a cloud architect, you should think creatively and think about ways you can implement elasticity in your application. Elasticity should be considered as one of the fundamental architectural design requirements or a system property to gain the advantages of underlying cloud's elasticity.

1) Application Elasticity using Cloud Foundry: There are actually two levels at which Cloud Foundry scales, whether automatically or not. The first is at the Cloud Foundry infrastructure level, e.g. how many app execution engines, how many request routers, how many cloud controllers, and how many services there are. The second level is at the individual application level and is primarily expressed in how many app execution engines are "running" the application (really, how many have the application loaded and are accepting requests from the request router).

The first level of scaling the Cloud Foundry infrastructure is the responsibility of the PaaS operator. The operator needs to monitor the load on the various servers and launch additional or terminate idle ones as appropriate. In particular, there should always be a number of idle app execution engines that can accept the next application or that can be brought to bear on an application that needs more resources. This level of scaling can be performed relatively easily manually or automatically in Right Scale. The app execution engines can be placed in a server array and scaled based on their load. The second level of scaling is the responsibility of each application's owner. The nice thing about the modularity of Cloud Foundry is that it exposes the necessary hooks to adding external application monitoring and scaling decisions. It is also interesting that Cloud Foundry in effect exposes the resource costs and lets the application owner decide how much to consume - and pay for. This is in contrast to other systems that make it difficult to limit the resources other than by setting quotas at which point an application is suspended as opposed to simply running slower [13].

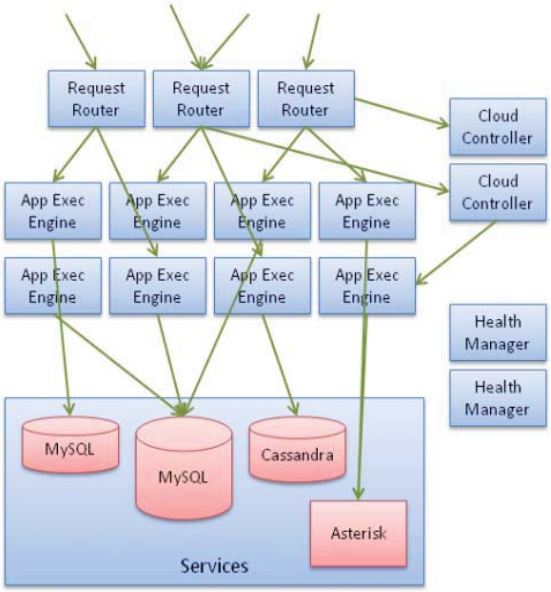


Fig. 10. Auto-Scaling in Cloud Foundry

2) Elasticity using Amazon Cloud: For example, Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. You can bundle the operating system, application software and associated configuration settings into an Amazon Machine Image (AMI). You can then use these AMIs to provision multiple virtualized instances as well as decommission them using simple web service calls to scale capacity up and down quickly, as your capacity requirement changes. You can purchase On-Demand Instances in which you pay for the instances by the hour or Reserved Instances in which you pay a low, one-time payment and receive a lower usage rate to run the instance than with an On-Demand Instance or Spot Instances where you can bid for unused capacity and further reduce your cost. Instances can be launched in one or more geographical regions. Each region has multiple Availability Zones. Availability Zones are distinct locations that are engineered to be insulated from failures in other Availability Zones and provide inexpensive, low latency network connectivity to other Availability Zones in the same Region.

Another example is, you can enable monitoring on an Amazon EC2 instance using Amazon CloudWatch2 in order to gain visibility into resource utilization, operational performance, and overall demand patterns (including metrics such as CPU utilization, disk reads and writes, and network traffic). You can create Auto-scaling Group using the Auto-scaling feature3 to automatically scale your capacity on certain conditions based on metric that Amazon CloudWatch collects. You can also distribute incoming traffic by creating an elastic load balancer using the Elastic Load Balancing service.

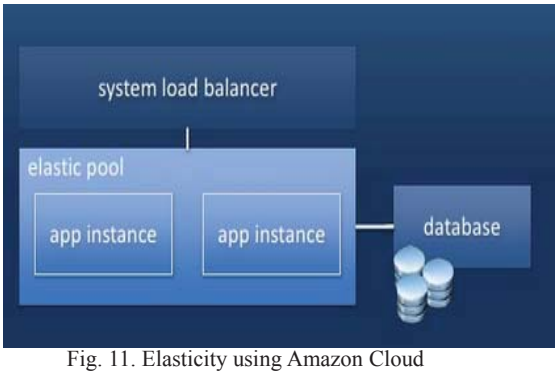


Fig. 11. Elasticity using Amazon Cloud

IV. Future Directions

The goal of Cloud applications is to abstract them from the underlying physical hardware. A cloud application should be able to plug into cloud like an electric utility (ex: washing machine) plugged into electrical socket without much bothering about the underlying details of electricity. In cloud, applications will continue to function even if the underlying

physical hardware fails or is removed or replaced. Applications will adapt themselves to fluctuating demand patterns by deploying resources instantaneously and automatically, thereby achieving highest utilization levels at all times. Scalability, Security, High availability, Fault-tolerance, Testability and Elasticity will be configurable properties of the application architecture and will be an automated and intrinsic part of the platform on which they are built.

However, it is not the reality today. Today, a cloud application architect can build applications to seamlessly deploy and run in the cloud by following the design considerations highlighted in this paper. However, cloud computing architectures will continue to evolve and as researchers, we should focus not only on building cloud applications but also on building tools, technologies and processes that will make it easier for developers and architects to plug in applications to any cloud easily and seamlessly.

V. Conclusion

This paper presents a study of various considerations architecting efficient cloud applications. By focusing on design concepts and best practices - like stateless applications, designing for failure, decoupling the application components, understanding and implementing elasticity, combining it with parallelization, and integrating security in every aspect of the application architecture - cloud architects can build highly scalable cloud applications.

There are various clouds that offers (Ex: CloudFoundry, AWS cloud, Azure etc.) highly reliable pay-as-you-go infrastructure services. As a Cloud application developer, the various design considerations highlighted in the paper needs to be considered. Also, it is advised that you experiment with various available services from open source and different vendors, learn from the work of others, build on the top, enhance and further invent cloud computing.

Acknowledgment

The authors of this paper are profoundly grateful to authors of various papers mentioned in the references section. Without going through those references, this paper would not have been possible. Also, the views and opinions expressed are purely based on the author's understanding of the Cloud Computing domain concepts.

References

[1] Developing cloud apps: What's different
http://www.infoworld.com/d/cloud-computing/developing-cloud-apps-whats-different-675

[2] Three Important Considerations for Cloud-Native Application Design: The Value of Designing for Resilience.

[3] Cloud Design Patterns: Prescriptive Architecture Guidance for Cloud Applications

[4] Amazon S3

[5] Architectural Strategies for Cloud Computing

[6] Architecting for the Cloud: Best Practices
http://jineshvaria.s3.amazonaws.com/public/cloudbestpractices-jvaria.pdf

[7] Planning the Migration of Enterprise Applications to the Cloud

[8] Introduction to Cloud Computing architecture

[9] Gartner Says Worldwide Public Cloud Services Market to Total $131 Billion
http://www.gartner.com/newsroom/id/2352816

[10] Cloud Computing and Enterprise Software Forecast Update, 2012

[11] MongoDB technical benefits

[12] Benchmarking Top NoSQL Databases. A Performance Comparison for Architects and IT Managers
http://www.datastax.com/wp-content/uploads/2013/02/WP-Benchmarking-Top-NoSQL-Databases.pdf

[13] Cloud Foundry architecture and Auto-Scaling

[14] MongoDB vs. SQL Server 2008 Performance
http://viethip.com/2011/08/24/mongodb-vs-sql-server-2008-performance/

DBMLE: Distance-Based Multi-Level Elliptic Routing Protocol for Ad hoc Networks

Kevin Joy Dsouza
Dept. of CSE
St. Joseph Engineering College, Mangalore
joydsouza33@gmail.com

Sujatha M
Dept. of CSE
St. Joseph Engineering College, Mangalore
sujatha_msk@yahoo.co.in

***Abstract*-- A critical challenge in the design of Ad Hoc networks is the development of an efficient routing protocol which provides high quality communication. The node's in MANET have limited communication resources such as bandwidth, buffer space, battery power etc. The resource constraints in MANET require the traffic to be properly distributed among the mobile host. The distance based multi-level elliptic routing protocol proposed in this paper is an efficient method of selecting the node as a next hop node, while communicating in a mobile Ad hoc network. The proposed method selects the node's which are orthogonal to the centroid. Here the node uses the position information to identify the nodes which lie in orthogonal direction. The node obtains location information from the GPS location service. The GPS location service fails to give accurate position information in constrained environments such as node not exposed to GPS satellites. In these scenarios, the node fails to get the location information. The error caused leads to locating the node to improper position. Therefore DBMLE protocol used in this paper uses communication signal strength along with the GPS information. The nodes keep track of neighbor node's position, velocity and signal strength information.**

Keywords—Ad hoc; MLE; DBMLE;

I. INTRODUCTION

Ad hoc is defined as "Arranged or happening when necessary and not planned in advanced" according to oxfords advanced learners dictionary. This gives an explanation of what Ad hoc networks are is to say networks set up on the fly for a special purpose. Furthermore ad hoc networks are usually such networks that are set up for one time occurrences such as conferences or military operations and as well as for organizational needs. This can be paraphrased into the following definition an Ad hoc network is a flexible and adaptive network with no fixed infrastructure.

Since Wireless Ad hoc Networks are inherently different from the well-known wired networks, it is a unique network topology where the network has no specific infrastructure. Since nodes will be moving, network can be formed and broken up any time. Thus some challenges raise from the two key aspects: self-organization and wireless transport of information. Because the nodes in a Wireless Ad hoc Network are free to move arbitrarily at any time. So the networks topology of MANET may change randomly and rapidly at times. This makes routing difficult because the topology is constantly changing and nodes cannot be assumed to have accurate information about the other nodes. In the worst case, we do not even know whether the node will still remain next minute, because the node will leave the network at any minute.

Bandwidth constrained is also a big challenge. Wireless links have significantly lower capacity as compared to wired network. Wireless links have lower throughput and involves Energy constrained operation. In MANET some or all the nodes may rely on batteries. For this kind of networks it's very important to consider and ensure the optimization of the energy conservation.

II. RELATED WORK

In multi-level elliptic routing presented in the paper [1] an efficient method to select the next hope is explained. In this scheme network is divided into regions and suppose communication has to be established then the method of drawing perpendicular from source to destination and moving along the perpendicular line in order to avoid network Centre is used. This reduces the maximum load of nodes in the network by avoiding the highly loaded network center.

This paper gives the different metrics to compare different routing algorithm: average load, expected maximum load, and maximum expected load of the nodes in the network. The average load of the nodes in the network is calculated by dividing the total number of packet transmissions in the network by the total number of nodes and hence minimum average load of the nodes is achieved using shortest-path routing .but in practical load of different nodes varies greatly and it is important to also study the maximum load of the nodes in the network.in this routing technique it also accomplishes k-BestNeighbor technique on greedy routing. In greedy routing technique at each hop neighboring node which reduces the Euclidean distance to destination is chosen.

In the paper [2] efficiency of Distance Based Routing every vehicle calculates the inter-vehicular distance between itself and its neighbouring vehicles the inter-vehicular distance is computed using formula as follows:

$$D = S*T \text{----------} (1)$$

Here D is the inter-vehicular distance, S is the velocity of vehicle and T is the propagation delay. Author states that inter-vehicular distance (D) is directly proportional to velocity(S) of the vehicle and propagation delay (T). Here selection of intermediate vehicle for the purpose of routing

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

directly depends on the relative speed of the vehicles and hence on the inter-vehicular distance .this is evident that inter-vehicular distance remains constant as long as no change in the velocity or direction of the vehicles. In this protocol every vehicle broadcasts position and speed information at time t and with respect to vehicle n. Based on the information received, vehicle n computes the inter-vehicular distance with respect to propagation delay. In addition this protocol makes use of digital map for every vehicle which provides entire detail of the road network such as coordinates of intersection. All the vehicles determine its initial position using GPS technology or from users and identifies its location in the digital map.

III. PROPOSED METHOD

In proposed distance based multi-level elliptic routing (DBMLE) we use a mechanism to push the nodes away or outside from the path of network center. Because it is assumed that network center is always heavily loaded. Since in a network there may be multiple regions we may need to push one or more nodes away from the path to network center. Here we use the map information from GPS to identify the nodes and calculate the distance between the nodes. This distance and signal strength information is always kept up to date for selection of best next hop in the path from source to destination.

The algorithm works as follows, we use two sets of parameters $r = [r_1, r_2, ..., r_k]$, where $0 \le r_1 \le r_2 \le ... \le r_k \le R$ and $f = [f_1, f_2, ..., f_k]$. The parameters r_i are used to divide the network area into disjoint regions. The functions f_i: $R \times R \rightarrow R$ are used to assign the intermediate points. Here network area is divided into two regions namely the inner region with distance r from the network centre and the outer region outside it. If source and destination nodes are in the inner region, the routing path is the same as in greedy routing. But in the case when either the source node or the destination node (or both) are in the outer region, the route goes through an intermediate point to avoid the network centre. Let *l* denote the line connecting the source node (src) and the destination node (dest). Also let *l'* denote the line that goes through the network origin and is perpendicular to *l*. If the intersection of *l* and *l'* does not lie between src and dest nodes on *l*, the path is again as in greedy routing. Otherwise we select an intermediate point *intPoint* on *l'* with distance $r_{intPoints}$ to the origin, defined by the parameter function *f*.

The decision of the function to use is made based on the regions where the source node and the destination node are located. In the first stage of the algorithm the number of intermediate points is determined. We find the largest *i* such that at least one of the source or the destination nodes is at distance greater than r_i to the network centre. The number of intermediate points is then calculated as 2^{i-1}.The coordinates of the intermediate points are calculated and stored in an array *intPoints* of size $2^i - 1$ using a recursive algorithm. First the middle element (i.e. *intPoints*[2^{i-1}]) is calculated as in Algorithm 3 using the given source and destination. Next the algorithm recursively fills in the first half of the *intPoints* array using the given source and *intPoints* [2^{i-1}] as destination, and similarly recursively fills in the second half of the *intPoints* array using *intPoints*[2^{i-1}] as source and the given destination.

Here the advantage of using map and distance function is for accuracy and selection best path for communication. The node uses the position information to identify the nodes which lie in orthogonal direction. The node obtains location information from the GPS location service. The GPS location service in some cases fails to give accurate position information in constrained environments such as node not exposed to GPS satellites. In these scenarios, the node fails to get the location information. The error caused leads to locating the node to improper position. Therefore DBMLE used in this project uses communication signal strength along with the GPS information. The nodes keep track of neighbor node's position, velocity and signal strength information's.

Following *Fig 1* shows how nodes are pushed away from the path of network Centre.

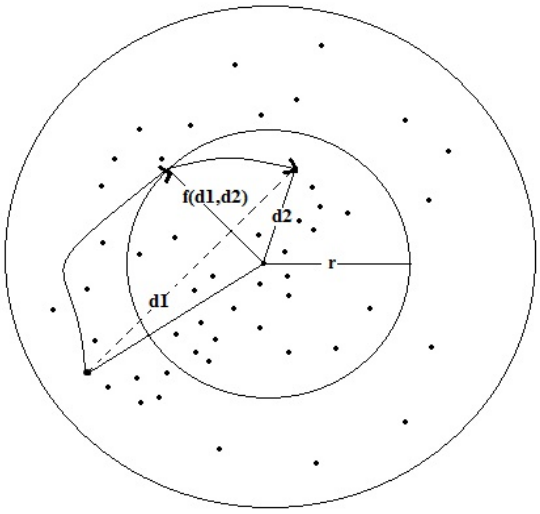


Fig 1: Path selection in DBMLE

ALGORITHM 1 : DBMLE Routing

---

function multilevelelliptic(*src, dest, r[], f[])*

path ← (*src*)

if dist(*src, dest*) ≤ Tr **then**

  path ← (*path, dest*)

  return *path*

end if

{else}

curr ← *src*

```
intPoints ← assignIntPoints( src, dest, r, f)
for i = 1 to size(intPoints) do
 {use greedy path to a node within the transmission range
    of intP oint[i]}
   while dist(curr, intP oints[i]) > Tr do
     curr ← greedy(curr, intP oints[i])
     path ← (path, curr)
   end while
end for
  {use greedy path to a node within the transmission range
    of dest}
while dist(curr, dest) > Tr do
curr ← greedy(curr, dest)
path ← (path, curr)
end while
path ← (path, dest)
return path
```

ALGORITHM 2: Assigns Intermediate Points

```
function assignIntPoints(src, dest, r[], f[])
k ← size(r)
level ← 0
numberIntPoints ← 0
for i = k downto 1 do
if dist(src, origin) > r[i] or dist(dest, origin) > r[i]
then
level ← i
numberIntP oints ← 2i − 1
break;
end if
end for
if level == 0 then
intP oints ← an empty array of size 1
intP oints[1] ← dest
else
intP oints ← an empty array of size numberIntPoints
calcIntPointsArray(src, dest, f[level], intPoints)
end if
return intPoints
```

ALGORITHM 3: calculate intermediate points

```
procedure calcIntPointsArray(p1, p2, f,A[])
    midIndex ← size(subArray)/2
    calcIntPoint(p1, p2, f,A[midIndex])
    if size(A) > 1 then
     p3 ← A[midIndex]
    calcIntPointsArray(p1, p3, f,A[1..midIndex − 1])
    calcIntPointsArray(p3, p2, f,A[midIndex + 1..end])
end if
procedure calcIntPoint(p1, p2, f, intPoint)
  (x1, y1) ← cartesianCoordinates(p1)
  (x3, y3) ← cartesianCoordinates(p2)
   l ← line going through (x1, y1) and (x3, y3)
  l' ← line going through origin and perpendicular to l
  (x2, y2) ← intersect(l, l_)
  if (x1 ≤ x2 ≤ x3 and y1 ≤ y2 ≤ y3) or (x1 ≥ x2 ≥
  x3 and y1 ≥ y2 ≥ y3) then
  (r1, θ1) ← polarCoordinates(x1, y1)
  (r2, θ2) ← polarCoordinates(x2, y2)
  (r3, θ3) ← polarCoordinates(x3, y3)
   r ← f(r1, r3)
  θ ← θ2
  intP oint ← polar2cartesian(r, θ)
  distance(x,y,intPoint)
else
  intP oint ← p2
  distance(x,y,intPoint)
end if
```

ALGORITHM 4: Calculate Distance

```
Procedure nodeId distance(x,y,intpoint)
if(locationinfoavailable)
{
   node_distance←Euclidiandistance(x1,y1,x2,y2)
```

```
If(node lies on intpoint)
    {
      Return nodeId
    }
  else
    {
     check← signal_strength
    node_distance←based on signal_strength
     if(node lies on intPoint)
       {
          return nodeId)
       }
    }
}
```

IV. EXPERIMENTAL RESULTS

The proposed routing protocol is simulated using network simulator 2(NS 2.34) [11]. The simulation environment is summarized in the Table I. As indicated in the TABLE I, the simulation was carried out for duration of seconds over an area of 1000 sq. meters with varying traffic from 5,10… to 25 connections.

TABLE I: SIMULATION PARAMETER

Simulation area(m x m)	1000 x1000
Simulation time(s)	100
Number of traffics	5,10,15,20,25
MAC layer protocol	IEEE 802.11
Transmission range (m)	250
Maximum velocity(m/s)	100
Hello interval	1ms

A. AVERAGE END TO END DELAY WITH NUMBER OF THE NODES

Results obtained for average end to end delay with respect to number of nodes for MLE and DBMLE are listed in the TABLE II. It is observed that the delay is less for DBMLE as the number of nodes increases. Average end to end delay of nodes is measured in time (ms) with respect to number of nodes.

TABLE II: AVERAGE END TO END DELAY WITH RESPECT TO NUMBER OF NODES

Number of nodes	MLE (msec)	DBMLE (msec)
5	140	118
10	200	150
15	275	240
20	360	200
25	340	195

B. AVERAGE END TO END DELAY WITH RESPECT TO SPEED OF THE NODES

Average end to end delays with respect to speed of the nodes for MLE and DBMLE protocol is recorded in the following TABLE III.

TABLE III: AVERAGE END TO END DELAY WITH RESPECT TO SPEED OF THE NODES

Speed(m/sec)	MLE (ms)	DBMLE (ms)
5	215	140
10	250	200
15	245	350
20	260	400
25	450	540

Average end to end delay for MLE and DBMLE are showed in the TABLE III. It is observed that delay in DBMLE is comparatively less as compared to the delay in the MLE. Delay is measured in time (ms) with respect to speed (m/s) of the nodes.

C. CONTROL PACKETS SENT WITH RESPECT TO NUMBER OF NODES

The number of control packets sent with respect to the number of nodes for MLE and DBMLE are showed in the following TABLE IV.

TABLE IV: CONTROL PACKETS SENT W.R.T SPEED

Control packets sent w.r.t speed (m/sec)	MLE	DBMLE
5	2700	1000
10	3800	6000
15	6300	7900
20	5000	4800
25	5500	5600

The TABLE IV shows the number of control packets sent with respect to speed (m/s). It is observed that DBMLE transmits packets more rapidly than as compared to the MLE as the speed increases.

D. CONTROL PACKETS SENT WITH RESPECT TO SPEED OF THE NODES

The following TABLE V shows the number of control packets sent with respect to speed of the node for MLE and DBMLE.

TABLE V: CONTROL PACKETS SENT W.R.T SPEED OF THE NODES

Control packets sent w.r.t. speed (m/sec)	MLE	DBMLE
5	17400	13000
10	17000	15000
15	16500	17000
20	18800	19000
25	17600	18000

It is observed that the number of control packets sent with respect speed (m/sec) increases as the speed of the nodes increases in DBMLE as compared to the MLE.

V. CONCLUSION

The proposed DBMLE protocol makes use of an efficient and effective method of selecting the nodes away from the path to network Centre and selecting best node to reach destination. This technique selects node's which are orthogonal to centroid. Advantage of this technique is that it uses GPS information and also keeps track of the node's movement information. All the nodes determine their initial position using GPS technology and identify its location in the map. Whenever there is a change in these parameters, the node will broadcast the hello message. The neighboring node which receives the hello message will update its routing table accordingly. It is obvious that GPS information fails to give accurate position information in constrained environment such as node not exposed to GPS satellite. DBMLE routing protocol uses communication signal strength along with GPS information. The node's keep track of neighbor node's movement and signal strength information. Hence the result obtained in this experimentation demonstrates the stability of DBMLE in a MANET environment.

REFERENCES

[1] Mohsen Eftekhari Hesari, Lata Narayanan, Jaroslav Opatrny, New Routing Algorithms to Balance Traffic Load *, IEEE WCNC 2011-Network,*pp 968-973.

[2] Ramakrishna M, "DBR: Distance Based Routing protocol for VANETs", 2011 International Conference on Network Communication and Computer (ICNCC 2011), March 2010.

[3] C. Perkins and E. Royer, "Ad-hoc on-demand distance vector routing," *IEEE workshop on Mobile computing Systems and Applications*, pp. 90–100, 2001.

[5] S. Durocher, E. Kranakis, D. Krizanc, and L. Narayanan, "Balancing Traffic Load Using One-Turn Rectilinear Routing," in *Proceedings of the 5th International Conference on Theory and Applications of Models of Computation*, 2008, pp. 467–478.

[6] I. T. Haque, T. Fevens, and L. Narayanan, "Randomized routing algorithms in mobile ad hoc networks," in *Proceedings of Sixth IEEE International Conference on Mobile and Wireless Communication Networks*, 2004, pp. 347–357

[7] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, pp. 153–157, 1996

[8] B. Karp and H. T. Kung, "Gpsr: greedy perimeter stateless routing for wireless networks," MobiCom 00: in *Proc. of the 6th annual international conference on Mobile computing and networking*, New York, NY, USA, pp. 243–254, 2000.

[9] J.-M. Zogg, GPS Basics, 2002.

[10]S. Singh, M. Woo, and C. Raghavendra, "Power-aware routing in mobile ad hoc networks," in *Proceedings of MobiCom '98*, 1998, pp. 181–190.

[11]Network simulator 2 (Ns2):http://www.isi.edu/nsnam/ns/

Experimental Comparison of Three Scheduling Algorithms for Energy Efficiency in Cloud Computing

Sudhir Goyal[1], Seema Bawa[2]
Department of Computer Science and Engineering
Thapar University
Patiala, India
mr.skgoyal@gmail.com, seema@ thapar.edu

Bhupinder Singh[3]
Progress Software Corporation
Hyderabad, india
bhupindersinghsaini@gmail.com

Abstract— **Nowadays, with the increased deployment of servers to facilitate high performance computing (HPC) for scientific and engineering applications lead to large consumption of energy. Cloud computing is a cost-effective solution, as it allows to host storage, computational and supported network services on a shared infrastructure of physical servers. However, the growing demand of cloud infrastructure among the IT companies is drastically increasing, by which data centers are drawing more energy. Energy efficient scheduling is one effective solution to streamline the resource usage as well as reduce the energy consumption. The proposed work in this paper demonstrates the resource allocation and makes an energy consumption analysis of Greedy, Round Robin and Power Aware Best Fit Decreasing (PABFD) scheduling algorithms on a private academic cloud. This paper provides an insight into the working of different scheduling scenarios for cloud computing and demonstrates the potential for the improvement of energy efficiency of PABFD algorithm under academic workload.**

Keywords— Cloud computing, high performance computing, energy efficient scheduling, resource allocation component;

I. INTRODUCTION

The biggest challenge facing modern cloud data centers are to operate with minimum operating cost while preserving quality of service to consumers. Data centers hosting cloud computing applications consume huge amounts of energy that ultimately contribute to high operational costs and carbon footprints to the environment. As energy cost is increasing, many organizations coming forward with a thought to operate their data centers in a "green" manner rather than sole concern of pure performance. Energy efficiency and 'Green' words are being used synonymously these days. Green Cloud computing is envisioned to achieve not only the efficient processing and utilization of a computing infrastructure, but also to minimize energy consumption [1].

A lot of research is being carried out to operate data centers in a "green" manner to conserve energy. The key areas of research in Green cloud computing are state-of-the-art energy efficient hardware, thermal aware scheduling, resource virtualization, green scheduling and resource provisioning polices. One effective way to reduce energy consumption is to switch-off unutilized servers through energy efficient scheduling. There are various scheduling and resource provisioning policies are used by private and public cloud vendors. Scheduling algorithms such as greedy, round robin and rank based scheduling used in private clouds like Nimbus, Eucalyptus and OpenNebula [2]. Apart from that, some authors [3] proposed algorithm such as Power Aware Best Fit Decreasing Algorithm (PABFDA) which were tested on CloudSim simulator to optimize energy efficiency parameter.

Novelty of proposed work is to study and analyze the energy efficiency of the existing scheduling algorithms on a private cloud for academic workload. This paper comprehensively demonstrate the resource allocation and power consumption of Round Robin, Greedy and PABFDA algorithms on a private academic cloud named ACA-CLOUD. Authors have opted Round Robin and Greedy algorithms as both of these two are widely used as cloud scheduling algorithms[2][4]. Being as third algorithm for analysis, PABFDA [3] has been selected, as it caters user VMs request with minimum performance loss. The basic workings of three scheduling algorithms are given as below.

Greedy Scheduling: In Greedy scheduling, all the VM's are placed on the first node till it can accommodate. If still VM requests are left then put to second available node and so on.

Round Robin Scheduling: Round Robin scheduling is wildly used as a load balancing technique in which cloud nodes are selected one after another until one is found that can accommodate the VM.

Power Aware Best Fit Decreasing Algorithm (PABFDA): PABFDA place a VM to a node that will increase least power consumption of overall data center. For VM migration, upper and lower CPU utilization's threshold of a host is specified.

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

When upper and lower cap of CPU utilization is violated, VMs are migrated from overloaded hosts to light loaded host so that performance problem can be avoided. An example to illustrate and analysis of these algorithms has been presented in section 4.

The remaining paper organized as follows: Section 2 discusses some of related and previous energy efficient scheduling algorithms. Energy consumption model is explained in Section 3. Section 4 contains experimental setup of a private cloud and in depth analysis on the working model of the three scheduling algorithms. Results are presented in Section 5. Finally, conclusions have been drawn in Section 6.

II. Related Work

A number of strategies have been proposed on cloud based VM placement and migration policies to maximize the profit of a data center by exploiting applications workload and the energy consumption. Buyya et al. [1] modeled VM allocation to physical machines as a multidimensional bin-packing problem in which the bins represent the physical machines and the items are VMs to be packed. To reduce the power consumption of overall data centre, the authors proposed Modified Best Fit Decreasing (MBFD) algorithm, in which all VMs are sorted in decreasing order of current utilization and mapped each VM to a host that increase least power consumption of a data centre. The work centered on architectural framework of energy efficient cloud computing. This work was further extended in [3]. Authors had explored the energy efficient resource management problem into two parts: first part was to provision and place new VM request on hosts using Power Aware Best Fit Decreasing (PABFD) algorithms, whereas the second part was to optimize the present VM allocation on cloud nodes. The proposed heuristic was migrating a VM from a host when CPU utilization of a server either exceeds upper utilization threshold level or goes below the lower threshold level. Minh et al. [5] had emphasized to more use of high performance and less energy consumption servers for reducing overall energy consumption of a data center. Authors proposed T-allocation heuristic where heavy load applications were placed to greater number of cores servers and keep light load applications to older servers having less number of cores. Since servers with smaller number of cores had a less chance of full load so it can be switched off. Their algorithm is more effective where heavy loaded applications run on older servers. Carbon/energy based scheduling policies were purposed in [6], which had exploited the heterogeneity across multiple datacenters based upon High Performance Computing (HPC) applications. Authors had taken a number of energy efficiency factors such as energy cost, carbon emission rate and workload which changed according to different geographical locations of data centers. Two energy-conscious task consolidation heuristics Energy-Conscious Task Consolidation (ECTC) and MaxUtil was presented in [7] , which aim to minimize power consumption by maximize resource utilization. Authors' greedy approach named as MaxUtil, made an effort to reduce energy consumption by consolidating as many tasks as it could to a VM. Further work was extended for large virtual clusters through energy-aware task consolidation (ETC) [8]. ETC minimized the energy consumption by the restricting CPU use below a specified peak threshold. ETC did this by consolidating tasks amongst virtual clusters.

Younge et al. [9] had shown through their work that the energy consumption does not increase proportionally as the number of processing cores increases. In another word, allocating virtual machines to minimum possible physical machines and switching off the others, consumes less energy than when equally allocating these virtual machines to all the physical machines. Authors had tested their power based scheduling on OpenNebula multi-core cluster. Further, it was concluded that proposed power based scheduling conserve 12% less of the system's power in comparisons of default non power aware scheduling policy of OpenNeubla scheduler. An energy-aware framework named as Green Open Cloud (GOC) to manage cloud resources had been proposed in [10]. Authors proposed an idea of energy conservation by switching off servers based upon users request prediction. With the previous feedback, next request reservation is predicted and decided to switch on or off the server based upon that prediction. Further, authors presented round robin and unbalance scheduling approaches under Basic, Balancing, On/off and Green scenarios. Ching-Chi et al. [11] had proposed two new strategies: Dynamic Round-Robin (DRR) and Hybrid for virtual machine deployment and migration among the computing nodes. Algorithm DRR reduce the power consumption of a data center by switching on minimum number of computing node. Authors work in [12], was providing the solution of scalability problem in managing energy consumption in large web server cluster. An algorithm, "the solution generation" addressed to the scalability problem in managing energy consumption through a hierarchical approach.

Most of the research work on energy efficiency focused on web servers and High Performance Computing (HPC) servers workload of a large data centers [3][6][13]. Both kind of workload is unpredictable as one could not predict in advance the web requests of users and their usage pattern. HPC workload is also depends upon the type of applications, duration and it varies each time. This work studies the behaviour of existing algorithms for energy consumption under the academic workload. Academic workload is stable and predictable as the student labs in Universities or institutes are conducted according to prescheduled time table.

III. Energy Model

Power consumption of server contributes to 75% of the total energy consumption of a data centre [5]. Thus, our objective is to minimize total power consumption of all the physical machines in a data centre. There are various parts in a physical machine that consume power such as memory, network interface, disk storage, CPU, power supplies etc. However there is a linear relationship between power consumption and CPU utilization [3][5][7]. It is shown as below [3].

$$P(U_{i.node}) = k \cdot P_{i.max} + (1 - k) \cdot P_{i.max} \cdot U_{i.node} \qquad (1)$$

$U_{i.node}$ is the CPU utilization of node i. $P_{i.max}$ is the maximum power consumed by ith node when the machine is fully utilized; k is the fraction of power consumed by the idle machine.

The energy consumption of a machine depends on its power consumption function over a period of time. Therefore, the total energy consumption E of a data centre is defined as follows.

$$E(T) = \int_0^T P_{Total}(t)dt \qquad (2)$$

IV. EXPERIMENTAL SETUP

For carrying out the proposed research, a heterogeneous Academic Cloud, named ACA-CLOUD, has been setup that consisting of three machines with data as given in Table 1. As a Virtual Machine Manager, Kernel-based Virtual Machine (KVM)[14] is installed on each node. Each virtual machine has 1 virtual CPU (VCPU), 1GB RAM and 35GB of secondary storage. Host1 and Host2 can accommodate maximum two VMs and Host3 can accommodate maximum 4 VMs. All machines are connected with 100Mbps Ethernet and 1 TB shared SAN-based storage. System CPU utilization sample have been taken after every minute. In our experimental setup, VM requests are coming from students for their own academic practical work with applications like SPSS, Internet Information Server, eclipse, browser, Netbeans, Java, C, C++. As a base operating system, ubuntu 12.4 [15] is installed on each machine.

TABLE I. POWER CONSUMPTION OF CLOUD NODES [16][17]

Nodes	Model	Cores	P_{idle} (watt)	P_{max} (watt)
Host 1 and Host 2	Intel Core 2 Duo 2.93GHz	2	39.3	78.9
Host 3	i5 2.5 GHz	4	39.8	106.8

A. Round Robin and Greedy Scheduling Algorithms VM Migration Strategies

In this section, Greedy and Round Robin scheduling algorithms share common strategies for VM migration which have been explained as below.

1) VM Migration module is triggered only at the start of each time slot of time table.

2) Source host has been chosen for VM migration which has minimum VM running. A host with minimum VMs has maximum possibility to switch off.

3) A host with maximum number of cores or virtual CPUs is selected as a destination for VM's migration. If a maximum number of cores host are unable to accommodate the VM, the next host with less number of core are chosen that can accommodate the VM.

B. Power Aware Best Fit Decreasing Algorithm's (PABFDA) VM Placement and Migration Strategies

This section describes PABFD algorithm's practical strategies as per requirement of our cloud model.

1) As the utilization is not specified by user in advance so we sort the VM in descending order according to the time duration of VM. A VM with greater time duration is allocated first and so on.

2) Under a migration policy, virtual machine is migrated from overloaded node as the upper cap of CPU utilization reaches at 85 percent.

C. Working Model of Greedy, Round Robin and PABFDA

This section describes the working model and power consumption analysis with the illustration of VMs request arrival scenarios given in Table 2.

TABLE II. VIRTUAL MACHINE REQUEST ARRIVAL SCENARIO

VM Arrival Time	Number of VMs request arrived	Virtual Machine Number (Duration minutes)
t_0 min	5	VM1(18 min), VM2(8 min), VM3(16 min), VM4(20 min), VM5(20 min)
t_{15} min	3	VM6(10 min), VM7(20 min), VM8(10 min)
t_{30} min	7	VM9(20 min), VM10(10 min), VM11(15 min), VM12(15 min), VM13(20 min), VM14(15)

1) Greedy Algorithm

Gantt chart of Greedy algorithm has been shown in (Figure 1) for input VM request pattern in Table 2. As shown in Figure 1, all the requested VMs from the queue at time t_0 are placed on Host1 and Host2 to the maximum extent it can accommodate. Being as the resource manager last time pointed at Host3 by allocating VM5 on it; still there is more room to accommodate more three VMs. So at time t_{15}, all the requested VMs i.e VM6, VM7, VM8 are placed on Host3. Similarly at time t_{30}, current host is pointed at Host3 so VMs VM9, VM10 and VM11 (maximum allocation 4 VMs) are mapped on Host3. As discussed in sub section IV A, at time t_{40} and $t_{45,}$ VM14 and VM13 are migrated from Host2 and Host1 to Host3. Hosts Host1 and Host2 switch to sleep mode. Figure 2 depicts the CPU utilization of three cloud nodes for greedy scheduling with the VMs workload as shown in Figure 1. Figure 3 presents corresponding energy profile of three cloud hosts for greedy scheduling.

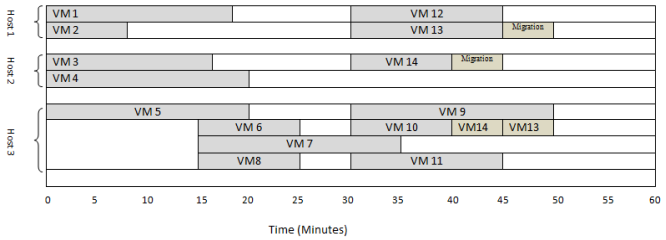


Fig. 1. Gantt chart for Schedule Output by Greedy

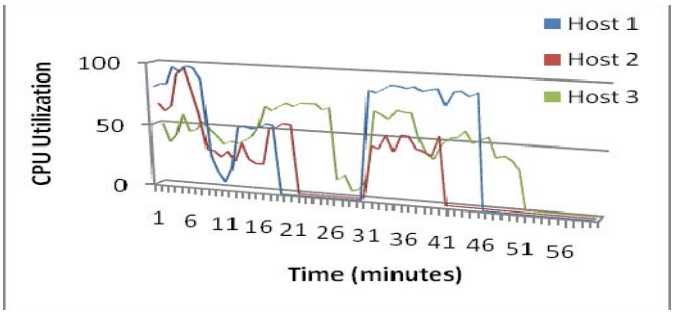


Fig. 2. CPU Utilization with Greedy Scheduling

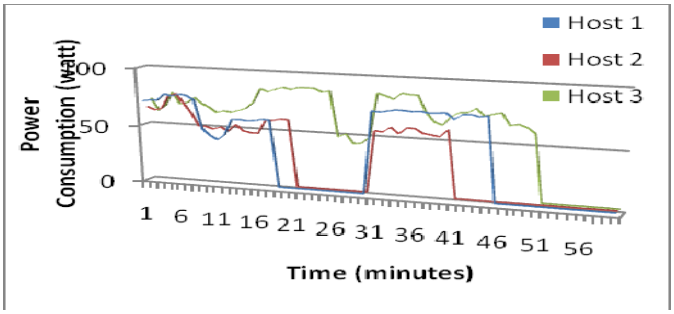


Fig. 3. Power Consumption with Greedy Scheduling

2) *Round Robin*

It prevents over usage of nodes to reduce overheating. As in figure 4, initially VM1, VM2 and VM3 are placed on Host1, Host2 and Host3 respectively. But as Host3 have more room to accommodate the VMs so VM1 and VM4 are migrated to Host3 and put the Host1 on low power mode. As per migration strategy, at time t_{20}, VM7 and VM8 are migrated from Host1 and Host2 to Host3 and switch the Host1 and Host2 into sleeping state. Similarly VM10 and VM13 are migrated from Host1 and Host2 to Host3 at time t35 and t45 respectively. Figure 5 presents the CPU's utilization of Round Robin scheduling for three hosts. Power consumption of three cloud nodes for this experiment is presented Figure 6. On comparison to the previous power consumption scenario, this one entails larger off-periods of Host1, so the power consumption is lower even though the cost of four migrations.

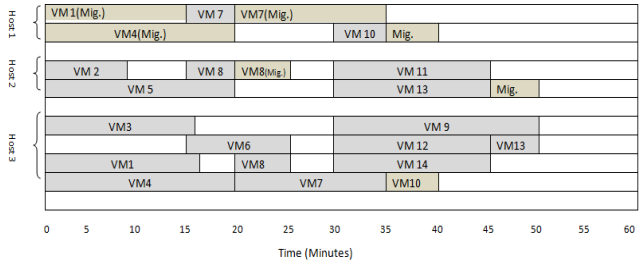


Fig. 4. Gantt chart for Schedule Output by Round Robin

3) *Power Aware Best Fit Decreasing Algorithm (PABFDA)*

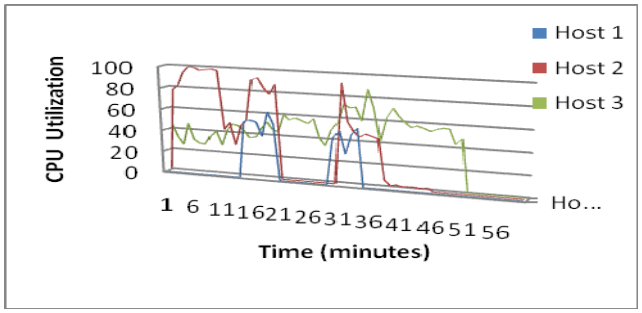


Fig. 5. CPU Utilization with Round Robin Scheduling

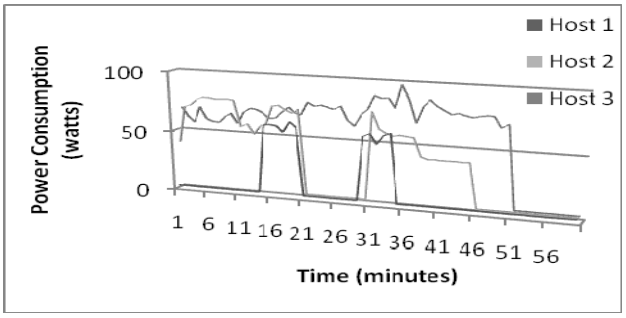


Fig. 6. Power Consumption with Round Robin Scheduling

As discussed in section (IV B), VMs are arranged in descending order of VM's execution time at time t_0. So VMs VM4, VM5, VM1 and VM3 are allocated on host 3 as shown in Figure 7. VMs are mapped on Host3 as it consumes less power in comparisons to aggregate power of Host1 and Host2. As the CPUs upper utilization cap of Host2 reached to 85 percent, VM10 is migrated from Host2 to Host1 under the migration policy. On our infrastructure, VM migration time is approximate 29 to 40 seconds. The CPU utilization and power consumption of the three cloud nodes is presented in Figure 8 and Figure 9 respectively. Compared to the previous power consumption scenarios, PABFD algorithm draws less power as aggregate power consumption of all the cloud nodes. Power consumption of accommodating four VMs on Host3 is less in comparisons to accommodate two VMs each on Host1 and Host2.

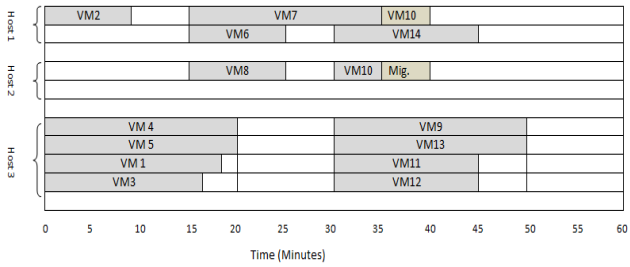


Fig. 7. Gantt chart for Schedule Output by PABFDA

V. RESULTS

To evaluate the Greedy, Round Robin and PABFDA, we have been conducted experiments with thirty different schedule

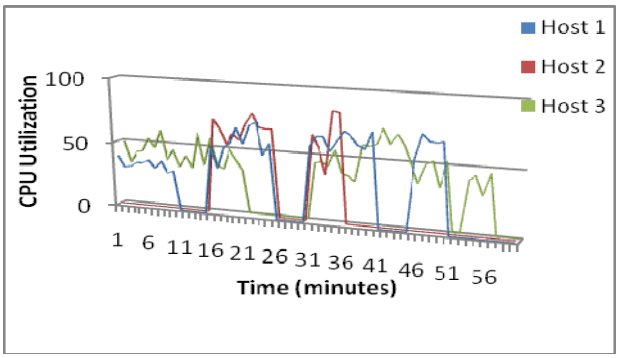


Fig. 8. Gantt chart for Schedule Output by PABFDA

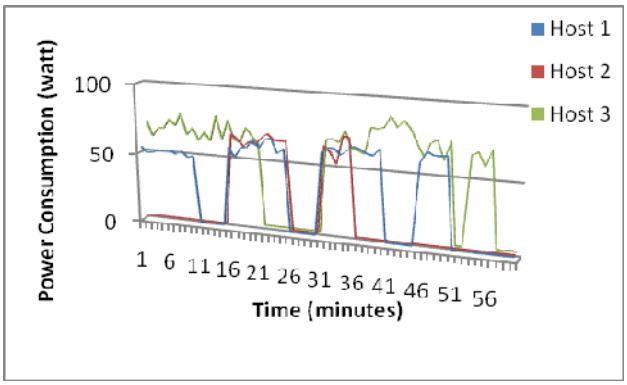


Fig. 9. Power Consumption with PABFDA

schedule patterns. Schedule samples are different in terms of number of different VMs requests and time duration. Figure 10 is plotted on the basis of the experiment results. Total power consumption of all the servers is basically dependent on a number of VM running, time duration and numbers of servers in sleeping state is kept by the algorithm. It comes to observation that PABFDA performs better in terms of power usage when number of requests are decreasing. To get statistically significant difference of Greedy, Round Robin and PABFDA algorithms, we conducted ANOVA and t-test on the collected data in regard to measure the power efficiency of algorithms. A null hypothesis which we have tested is that the mean power consumption by all the three algorithms is same. The results of ANOVA analysis reported in Table 3. Null hypothesis cannot be rejected at 5 percent significance level. This implies that statistically, power efficiency of three algorithms is same.

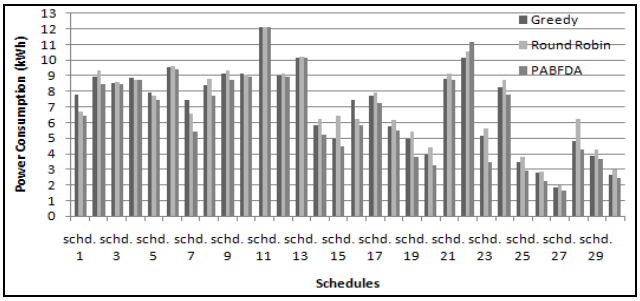


Fig 10: Power consumption of all the *all the nodes (Greedy, Round Robin and PABFDA)*

TABLE III. ANOVA TEST FOR THREE SCHEDULING ALGORITHM

Groups	*Count*	*Sum*	*Average*	*Variance*
Greedy	30	209127.03	6970.9	6897541.3
Round Robin	30	214834.43	7161.14	6351144.1
PABFDA	30	194647.17	6488.23	8126062.3

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	7219626.23	2	3609813.1	0.506646	0.604279	3.101296
Within Groups	619867683.2	87	7124915.9			
Total	627087309.4	89				

The results of ANOVA was not as per expectation and under the impression that one of the algorithm is significantly different from other two algorithms, we conducted t-test, where we compare the mean power consumption of one algorithm with mean power consumption of another algorithm. The null hypothesis is that the mean power consumption of two algorithms are same. The results of mean power consumption of Greedy algorithm with Round Robin are reported in Table 4. The results shows that two algorithms are essentially same in terms of power efficiency. In this case, the null hypothesis cannot be rejected at 5 percent significance level. However, when we performed t-test to measure power efficiency of Greedy with PABFDA and Round Robin with PABFDA, the results are found significant at 0.05 levels. The results of Greedy with PABFDA and Round Robin with PABFDA are reported in Tables 5 and 6 respectively. T-test results in Table 5 and 6 clearly indicate the competent power saving capability of PABFD algorithm in comparison to Round Robin and Greedy.

TABLE IV. T-TEST: PAIRED TWO SAMPLE FOR MEANS (GREEDY & ROUND ROBIN)

	Greedy	*Round Robin*
Mean	6970.901	7161.147844
Variance	6897541.299	6351144.118
Observations	30	30
Pearson Correlation	0.976984152	
Hypothesized Mean Difference	0	
df	29	
t Stat	-1.853848504	
P(T<=t) one-tail	0.03697915	
t Critical one-tail	1.699126996	
P(T<=t) two-tail	0.0739583	
t Critical two-tail	2.045229611	

TABLE V. *T-TEST: PAIRED TWO SAMPLE FOR MEANS (GREEDY & PABFDA)*

	Greedy	*PABFDA*
Mean	6970.901	6488.239
Variance	6897541	8126062
Observations	30	30
Pearson Correlation	0.979567	
Hypothesized Mean Difference	0	
df	29	
t Stat	4.429109	
P(T<=t) one-tail	6.17E-05	
t Critical one-tail	1.699127	
P(T<=t) two-tail	0.000123	
t Critical two-tail	2.04523	

TABLE VI. *T-TEST: PAIRED TWO SAMPLE FOR MEANS (ROUND ROBIN AND PABFDA)*

	Round Robin	*PABFDA*
Mean	7161.147844	6488.239068
Variance	6351144.118	8126062.281
Observations	30	30
Pearson Correlation	0.978425544	
Hypothesized Mean Difference	0	
df	29	
t Stat	5.692562902	
P(T<=t) one-tail	1.85679E-06	
t Critical one-tail	1.699126996	
P(T<=t) two-tail	3.71359E-06	
t Critical two-tail	2.045229611	

measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

VI. CONCLUSION

Energy efficient scheduling is an important approach to streamline the resource usage and in turn improve energy efficiency. This paper tried to explain the resource allocation and power consumption under different scheduling scenarios. The experiment results reveal that PABFD scheduler conserve significant energy; specially when number of VMs request are less in comparison to Round Robin and Greedy Scheduler. Many cloud computing organizations are focusing on standard issues like performance, fault tolerance, load balancing etc. But it is also true that to minimize the data centre's operational costs and carbon footprints to the environment, there is an urgent need to work on energy efficient framework, system architectures and green scheduling algorithms for cloud computing environments.

REFERENCES

[1] R Buyya, A Beloglazov, J Abawajy, "Energy-efficient management of data center resources for cloud computing A vision, architectural elements, and open challenges" In Proc of the 2010 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2010), Las Vegas, USA

[2] A. Nathani, S. Chaudhary, G. Somani "Policy based resource allocation in IaaS cloud," Future Generation Computer Systems, vol 28, no. 1, pp. 94-103,2012

[3] A. Beloglazov, J. Abawajy, R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," Future Generation Computer Systems, vol. 28, no. 5, , pp. 755-768. 2012

[4] Nimbus Project Docs (2013) Retrieved June 15, 2013 from http//wwwnimbusprojectorg/docs/2101/changeloghtml#2101

[5] Q. D. Minh, M. Federico, S. Domenico, R. Giafreda, "T-Alloc A practical energy efficient resource allocation algorithm for traditional data centers," Future Generation Computer Systems, vol 28, no 5, 2012, pp. 791-800.

[6] S. K. Garg, C. S. Yeo, A. Anandasivam and R. Buyya "Environment conscious scheduling of HPC applications on distributed cloud-oriented data centers," Journal of Parallel and Distributed Computing, vol 71, no.6, pp. 732-749

[7] Y. C. Lee, A. Y. Zomaya, "Energy efficient utilization of resources in cloud computing systems, " *The Journal of Supercomputing*, Vol. 60, Issue 2, pp. 268-280, May 2012

[8] C. Hsu, K.D. Slagter, S. Chen, Y. Chung "Optimizing energy consumption with task consolidation in clouds" Information Sciences,Vol 258,pp : 452-462, 2014

[9] A. J. Younge, G. Laszewski, L. Wang, S. Lopez-Alarcon and W. Carithers "Efficient Resource Management for Cloud Computing Environments," In Proc. of the IEEE International Green Computing Conference (IGCC), Chicago, IL, pp 357-364 , 2010

[10] L. Lefèvre, A-C. Orgerie, "Designing and Evaluating Energy Efficient Cloud," *Journal of Super Computing*, Vol. 51, No. 3, pp 352-373, 2010.

[11] L. Ching-Chi, L. Pangfeng, W. Jan-Jan "Energy-efficient Virtual Machine Provision Algorithms for Cloud Systems," In Proc. of the 4th IEEE International Conference on Utility and Cloud Computing (UCC), pp. 81-88, 2011

[12] L.S. Sousa, J.C.B. Leite, O. Loques "Green data centers Using hierarchies for scalable energy efficiency in large web clusters" Information Processing Letters, vol. 113, no. 14, pp. 507-515, 2013

[13] A. Beloglazov, R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers, " *Concurrency and Computation Practice and Experience*, vol 24, no. 13, pp 1397-1420, 2012.

[14] Kernel-based Virtual Machine (2013) Retrieved on June 15, 2013 from http//wwwlinux-kvm.org/page/Main_Page

[15] Ubuntu operating system (2013) Retrieved on June 15, 2013 from http//wwwubuntu.com/

[16] I. Gavrichenkov , CPU Benchmark (2010) http//wwwxbitlabscom/articles/cpu/display/cpu-benchmark-mainstream_10html#sect0

[17] I. Gavrichenkov , Power consumption of Intel Core i5 processor (2011) http://wwwxbitlabscom/articles/cpu/display/core-i5-2500-2400-2300_10.html

MultiPaaS - PaaS on Multiple Clouds

Shashank Sahni
Computer Engineering
International Institute of Information Technology
Hyderabad, AP - 500032
Email: shashank.sahni@research.iiit.ac.in

Vasudeva Varma
Search and Information Extraction Lab
International Institute of Information Technology
Hyderabad, AP - 500032
Email: vv@iiit.ac.in

***Abstract*—Current PaaS solutions use an underlying IaaS provider and restrict themselves to a few geographical regions. They operate in one/two zones only and use others for failover or disaster recovery. In this paper, we present the design of MultiPaaS, a PaaS solution which runs on multiple cloud/IaaS providers and leverages the combined global span. Unlike traditional scaling techniques where app servers are scaled up or down in a specific region or data center, we present a design to perform it on a global scale. Origin of the request burst is identified and instances are booted in a data center closest to it. It ensures database persistence and cross-region scalability via a combination of replication techniques. We identify common features available across all IaaS providers and use that as a provider interface to ensure interoperability, but use provider specific extensions to leverage unique offerings and benefits.**

I. Introduction

Cloud providers have simplified operations of developers and system administrators by enabling flexible resource provisioning and programmatic access. Users pay for what they use, making experimentation and deployment cheaper. This has resulted in rapid cloud adoption by small companies and startups who cant afford costs incurred from an on-premise infrastructure.

The cloud computing ecosystem comprises of multiple service offerings in the form of as a Service model. One of them, Platform as a Service(PaaS), has gained immense popularity among developers for quick deployment of web services. For a single request of developer, PaaS handles system provisioning, base configuration and application deployment. This enables a developer to not waste time or be concerned about configuration, deployment or scaling application. Its automatically and optimally handled by PaaS, making it a popular application hosting solution among developers.

With so much involvement on top of the stack, public PaaS providers prefer offloading infrastructure management by using an Infrastructure as a Service(IaaS) provider. This provides them the benefit of reduced operational overhead of managing physical infrastructure, reduced upfront costs and infinite scaling.

Most of the popular PaaS providers - Docker Inc.(formerly DotCloud)[1], Heroku[2], Pantheon[3], EngineYard[4] etc. operate on present IaaS providers, Amazon Web Services(AWS)[5] being the most popular.

Popular PaaS platforms are primarily used for hosting web services and applications. These are mainly user applications which either provider a web UI or act as a backend service for a mobile application. Both target best user experience demanding the application to be scalable and efficient.

With hosted applications targeting more and more users, its important to have systems closer to the end users. For PaaS a global footprint is imminent. No single IaaS provider can cover the entire globe efficiently. But since PaaS solutions already leverage a pre-existing IaaS, by combining multiple providers, a PaaS can offer a bigger global span.

Interestingly, except EngineYard[4], all the PaaS providers listed above have restricted their services to one/two regions in U.S. only. But similar to AWS itself, Engine Yard relies on the options provided by the user for region, db replication and instance selection. To effectively leverage a global footprint, the PaaS should smartly handle scaling and geo-balancing requirements.

This can be very helpful in providing optimal delivery globally. Similar to CDN, apps can be spawned in datacenter closer to the traffic origin. With regions come the benefit of automatic failover handling and disaster recovery. Due to the global coverage, organizations which are restricted to operate inside the country due to government regulations will be able to use the PaaS too.

AWS provides spot instances at very cheap rate with no guarantee whereas DigitalOcean[11] provides cheaper monthly fee for on-demand instances. Major IaaS providers like AWS and Rackspace[6], provide access to a lot of vendor products via their marketplace. Such a hybrid environment brings opportunity to utilize specific features and benefits offered by each vendor.

In this work, we present the design of MultiPaas, a multi cloud Platform as a service solution. It aims to address the primary issues of building and running a PaaS on multiple cloud/IaaS providers spread across globe and leverage the benefits offered by them.

The rest of the paper is structured as follows - In section 2 we talk about similar solutions provided by open source softwares and commercial vendors and discuss their shortcomings. In section 3 we briefly discuss the design of PaaS and the applications which run on them. Section 4 talks about challenges in building a multicloud platform. In section 5 we present the design of MultiPaas. Section 6 explains how scaling and load balancing are handled by MultiPaas. Section 7 proposes implementation specifics and experiment setup. Section 8 concludes the paper and showcases the vision for the project.

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

II. Related Work

The ability to use multiple cloud provider has been a challenge since the advent of commercial cloud providers. Plenty of solutions have been developed to tackle different aspects of multicloud environment.

Multicloud libraries exist in almost every programming language to enable the development of applications which leverage multiple cloud providers. jClouds(JAVA)[12], libcloud(Python)[13], fog(ruby)[14], pkgcloud(NodeJS)[15] are a few popular multicloud libraries.

Companies like RightScale[16] and Scalr[17] facilitate the management of multiple cloud providers from a single interface. OnApp[18] will be releasing their provider marketplace via cloud.net[19], providing both a web UI and API to access multiple providers worldwide.

These offerings showcase the demand and requirement of multicloud solutions. All of them provide accessibility to multiple providers from a single interface, but none provide an end to end solution of deploying and scaling an application across providers globally. The tasks of when and where to deploy and scaling up/down are entirely dependent on the user. Amazon Web Services(AWS) being the leading IaaS provider spans 5 continents via 8 different regions, making load balancing across data center and geographic regions an important aspect for their major customers. AWS Elastic load balancer[21], allows application workload distribution across availability zones(data centers in the same region). Based on the instance resource utilization, ELB is also capable of scaling instances on the fly.

AWS DNS offering, Route53[20], enables geo-load balancing via latency based distribution[29]. The endpoint closer to the requesting client(based on low latency) is sent as resolved IP.

These services although provide geographic load balancing and scaling, the latter is strictly regional to mitigate failovers and only operates on a pre-deployed infrastructure. The task of spawning instances in new locations to reduce latency is to be performed by the administrator. Furthermore, this solution integrates very tightly with AWS offerings and results in vendor lock-in.

Our work architects a self scaling Platform as a Service offering which uses multiple cloud providers to span vast geographical regions. Application scaling is automatically handled by identifying increasing load from a region and booting instances in a data center nearest to the end user.

III. Design of PaaS

Before we discuss the design of MultiPaas and the challenges involved, it would be beneficial to have a preliminary understanding of how a PaaS works and the popular applications hosted on it. This would help us gain a better understanding of the shortcomings and challenges when the design is modified to work on multiple cloud providers.

A. Working of a PaaS

A generic platform as a service offering has the following primary components

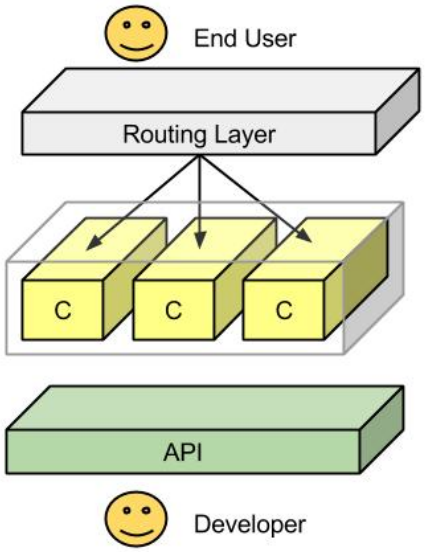


Fig. 1. Generic Paas Architecture.

- **Controller** - Controller node handles lifecycle of application on the platform. It exposes an API which is utilized by the developer to operate PaaS.
- **Routing/Load balancers** - Load balancers act as the web endpoint for the incoming requests to applications. The load balancers route the request to the appropriate application based on the hostname used in the request. They are also capable of distributing request load based on the instance health/usage and may even scale if the underlying application servers are exhausted.
- **Application Servers** - Application servers are usually virtual machines or containers which host the clients application. These servers can also be used to host Database required by the application.

B. Application Architecture

Common applications hosted on PaaS are three tier web applications[30] comprising of presentation layer, application/logic layer and data layer. The presentation layer is the web interface available to the user. Logic layer hosts the application code and data layer usually hosts the data store required by the application in logic layer. Presentation layers are usually scaled via Content Delivery Networks which can be easily integrated with a web application. The response time and user experience for the presentation layer are dependent on the performance and efficiency of the underlying layers. Hence, with respect to MultiPaas, we will be focus on scaling application layer and data layer.

IV. Challenges

PaaS running on multiple providers globally provides a great opportunity for scaling. Based on the origin of the incoming requests, application servers can be placed closer to the users, improving the delivery time and performance. This, however, introduces few problems.

1) **When to Scale?** With requests originating from different locations, PaaS should be able to quickly identify when scaling is needed. If it responds in haste to a short term request burst, scaling will go in vain in addition to extra cost incurred. If response is too slow, the performance will take a hit.
OpenShift, a PaaS solution from RedHat, uses number of requests as a measure to decide whether to scale or not. But, with the ability to spawn/scale applications in multiple locations worldwide, performance isn't the only criteria for deciding whether to scale or not. If a majority of the requests are originating from halfway around the world, one can gain significant benefits by placing the app servers in a data center closer to that region. Identifying a suitable datacenter closer to the location of requests origin is needed.
2) **Where to Scale?** In order to scale geographically, it is very important to correctly identify the origin of the incoming requests.
Cloudflare, a popular hosting solution, leverages anycast for geographically optimized delivery. Anycast, a routing scheme, provides the benefit of routing request to the closest server among a cluster of systems, all listening to the same IP. In CloudFlare, this has been a result of configuring network equipments across their data centers, coordinating with upstream providers etc. But control over hardware isnt a privilege enjoyed by everyone. PaaS providers leverage underlying IaaS, making Anycast not a viable option.
3) **Route users to the nearest servers/node** - Traditionally, webmasters have been using CDN providers extensively to deliver content to the end users in the most efficient way. This is expensive and is applicable to static content only. In addition to this, a well integrated DNS + Hosting solution can provide the benefit of directing users to the nearest node scaling web apps. CloudFlare has been doing the same for a while.
AWS DNS service Route53 provides a latency based name resolution feature for their services - EC2 instances and Elastic Load balancers. This feature ensures that requesting user is responded with IP address closest to it, based on the latency. This service is tightly integrated with AWS restricting PaaS to one provider only.
4) **Variance with Multiple Providers** Multiple providers introduce multiple services and access interfaces. This causes deployment and architectural changes while trying to provide same service transparently across providers. AWS and Rackspace provide block storage as a service for persistent storage whereas linode[7], digitalocean and other virtual hosting provide persistent storage out of the box. APIs and entities/features of each service differ among providers - IOPs can be controlled in AWS EBS but not in OpenStack. Although there exist plenty of unified API solutions - jClouds(java), libcloud(Python), fog(Ruby) etc. all of them abstract common features causing exclusive attributes of a cloud provider useless. Such differences need to be considered while designing a provider agnostic solution which is capable of leveraging everyones strength.
5) **Handling data across regions and providers** After identifying the increase in load metrics, most of the PaaS solutions scale by booting a calculated number of nodes of a specific configuration in the same region. Since the service has been operating in the same region, the data resides in the vicinity. For Amazon Web Services, even if the apps are booted across multiple availability zones(AZ) in the same region, due to good connectivity between AZs in a region, the latency and performance degradation is very low.

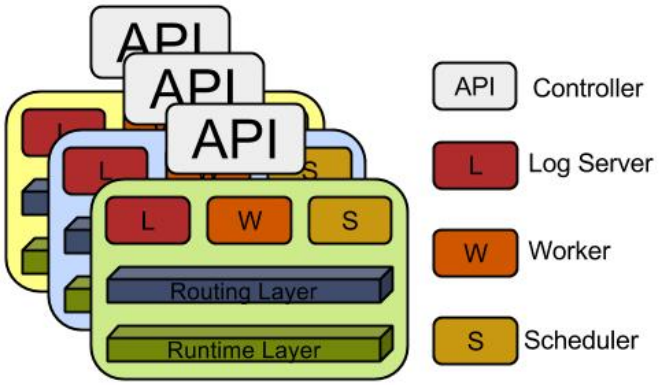


Fig. 2. Proposed MultiPaas Architecture.

This however isnt the case with vendor agnostic geo scaling. Data access over WAN can lead to noticeable performance degradation and data corruption if proper measures arent in place. With multiple instances of an app running across different providers globally, maintaining a persistent database with reasonable performance is a challenge.

V. Designing MultiPaaS

The design is motivated from some of the major PaaS vendors and opensource PaaS solutions Heroku, CloudFoundry[8], OpenShift[9] and Deis[10]. Due to the architectural similarity and flexibility provided by Deis, we decided to implement our design on it.

To avoid complexity in the architecture, we've designed multicloud PaaS as a set of multiple regional PaaS deployments. Every regional PaaS deployment runs a common set of services in their respective regions and export API for management, accessible over region specific hostnames. The design is analogous to how IaaS providers operate - AWS, Rackspace etc.

Note that, since MultiPaaS can be deployed across different IaaS providers, region isnt specific to any one of them. The region can be granular upto city level.

Components of the Architecture/Design Following services comprise a regional PaaS.

- **Routing Layer** - This layer consists of servers responsible for routing requests smartly to appropriate containers running the concerned application code.

- **Runtime Layer** - This layer consists of servers hosting the containers which host and run clients applications. The layer also hosts data specific containers which host database required by the application. The database isnt shared among applications and one container hosts only one instance of it.
- **API** - Each regional PaaS installation has an API endpoint. This endpoint is exposed by the controller service of the PaaS and is accessible from a region specific endpoint. This enables a client to start/scale/stop an app in a region via an API call to the correct endpoint.
- **Log Server** - Incoming requests for all servers in the runtime layer are logged centrally at this server.
- **Worker(s)** - This node(s) is responsible for processing log data periodically to update the app, request-freqency per region mapping.
- **Scheduler** - This server is responsible for taking load balancing and scaling decisions. Once a decision is made, it sends scaling request to the API endpoint of the region.
- **Applications** - Apps or specifically web apps are client code which would be hosted on the PaaS. Each virtual machine in the runtime layer would be hosting one application only. The routing layer would be responsible for directing requests to appropriate virtual servers.

VI. Scaling and Routing

A. When to Scale?

OpenShift uses number of requests as the criteria for scaling. If the number of requests for all the nodes reaches 90% of this threshold, new nodes are spawned. One can use the same mechanism for region based requests. For each application if the average number of incoming requests from a region crosses 16/second/server, then boot an instance in that region.

This approach handles both region specific and geographical scaling but makes decisions naively. If this sudden surge in traffic lasts only a few mins, a scheduler using this approach would have booted an instance in the concerned region, wasting resources. The decision making can be optimized by considering average number of connection over a period of time.

Similar to system load representation of unix systems by uptime command, MultiPaaS keeps average number of concurrent requests for 1, 5 and 10 mins for every region. If at any point the average value for a different region is more than 16 in all three cases, a new app server/container is booted in that region.

B. Where to Scale?

In order to identify the region, the IPs of the incoming request need to be mapped to a geographic region. We are using Maxminds GeoIP database/API[24] to accomplish this. Each query for an IP returns the country, region and latitude/longitude of the origin. We use the geospatial distance to find the nearest data center to the origin of the request.

This requires us to manually maintain geospatial location of every datacenter/region for each provider. Since every provider has few regions each covering multiple countries, the total is handful, making this data manually maintainable.

The worker nodes parse the web request logs collected from the load balancers to calculate request frequency per region per app. Since the typical http log format, Fetch the geo location for the client IP. The result is first looked up in a cache datastore. In case of a miss, its requested using the GeoIP API. Based on the location fetched, find the closest service provider geographically. Update the app, provider-frequency mapping for that minute. At the end, the process calculates the frequency of the requests per region per app for over a period of 1, 5 and 10 minute.

The scheduler will loop over this processed data and make scaling decisions. If the threshold is reached for the same region as the regional PaaS itself, scale horizontally in the same region.

If the threshold is reached for a different region, make an appropriate call to the API server to spawn app instances in a datacenter of that region.

C. Route users to nearest node/server

Major CDN providers like CloudFlare[28] use anycast to optimize content delivery by sending requests to the closest node. Anycast routes the request to the closest server among a cluster of systems, all listening to the same IP. This requires control over networking equipments and involvement with upstream ISPs which isnt possible for most of PaaS providers.

We solve this issue through DNS. Initial DNS protocol only shows the IP of the requesting dns client which is mostly ISPs DNS server for recursive name resolution. The new edns-client[31] support added to DNS protocol allows DNS server access to the requesting clients IP instead of only ISPs DNS server IP. This enabled the server to identify the location of the end user and resolve the hostname to the geographically closest server.

IPs of the new nodes - result of scaling, will be added to the DNS server A record by the scheduler. A new DNS request from the end user would automatically direct the request to the nearest nodes.

D. Variance with Multiple Providers

We are building a vendor agnostic PaaS that scales globally. In order to ensure biggest global footprint, integration with multiple IaaS providers is required. With multiple providers come differences in service offerings, API, price, performance etc. In order to ensure seamless integration and transparency to customers, MultiPaaS will use minimal services which are common across most of the providers - compute and block storage. Compute nodes will be used to host the application and block storage to host the persistent database.

Note that, similar to other PaaS environments no provider specific credentials are required from the user. All provider accounts are specific to MultiPaaS only.

E. Data Persistence Globally

Popular providers like Heroku provide database for the hosted web applications. With multiple instances of the web service running across different providers globally, maintaining a persistent database with appropriate performance is a challenge.

In order to provide geographical access along with data persistence, database will be deployed in master-master or master-slave clusters. The cluster will scaled along with the application.

VII. Prototype Proposal and Evaluation

We propose to implement the prototype by extending Deis[10], free and opensource PaaS offering supported by OpDemand. The platform is written in Python and uses docker containers to host applications. It is capable of supporting various hosting platforms via pluggable modules making it vendor agnostic.

For the geo capable DNS, we will be using gslb[23]. It also uses maxminds GeoIP database, ensuring that the nearest provider would be same for both MultiPaas and gslb.

The geographic location data of the cloud providers data centers would be stored in MongoDB database. The geospatial function, geonear[25], provided by Mongodb would allow us to query the nearest datacenter for each requesting IP quickly.

For the prototype, we will be restricting database to Perconas MySQL[26]. The database will be hosted as a master-master cluster on the block storage volumes and will be accessible over network to the application instances. A cluster would ensure easy scalability of the database across regions.

A. Evaluation

For MultiPaaS evaluation, we will host a sample web application on it which exposes only REST API . The API would provide a mix of read and write queries. This would enable the performance testing of the application based on the response time for different requests.

For the setup, we will be using AWS and Rackspace. These two are the leading cloud providers and together provide a wider global span. As per our design, MultiPaaS will be running on each region of these providers as multiple independent PaaS instance. The sample application will be uploaded to one of the regional PaaS and started.

A sample of servers will be booted in each region running a test script which would send requests to the applications API. This will help simulate global access. MultiPaaS will scale applications globally based on the request burst. The test script would record the response time and server for each API request.

The experiment would help us gain the following insights.

- Load incurred by the systems during the pre-scaling phase.
- Time taken to scale an application in a different geographical region.
- Time taken for the application load to settle down after scaling.
- A better estimate of the number of requests/app/region which should be used to make scaling decisions.

VIII. Conclusion and Future Work

We present a design of MultiPaaS, a Platform as a service offering which is capable of running on multiple cloud providers. We discuss the details of scaling an application across multiple providers enabling geographic load balancing. We further discuss implementation specifics on Deis, an open-source PaaS written in Python.

Our next target is to implement the solution on Deis and perform tests to evaluate the platform. This would help us gain insights into the practical bottlenecks incurred by the current design, helping us further improve it.

References

[1] ”Docker - Build, Ship, and Run Any App, Anywhere.” 2004. 23 Jun. 2014 http://www.docker.com/

[2] ”Heroku — Cloud Application Platform.” 2010. 23 Jun. 2014 https://www.heroku.com/

[3] ”Pantheon — The Professional Website Platform for Drupal ...” 2010. 23 Jun. 2014 https://www.getpantheon.com/

[4] ”Engine Yard — Cloud Application Management Platform.” 2010. 23 Jun. 2014 https://www.engineyard.com/

[5] ”Amazon Web Services (AWS) - Cloud Computing Services.” 2005. 23 Jun. 2014 http://aws.amazon.com/

[6] ”Rackspace: The Leader in Hybrid Cloud.” 23 Jun. 2014 http://www.rackspace.com/

[7] ”Linode: SSD Cloud Hosting.” 2005. 23 Jun. 2014 https://www.linode.com/

[8] ”Cloud Foundry.” 2011. 23 Jun. 2014 http://cloudfoundry.org/

[9] ”OpenShift by Red Hat.” 2012. 23 Jun. 2014 https://www.openshift.com/

[10] ”Deis — Your Paas. Your Rules.” 2013. 23 Jun. 2014 http://deis.io/

[11] ”DigitalOcean: SSD Cloud Server, VPS Server, Simple Cloud ...” 2012. 23 Jun. 2014 https://www.digitalocean.com/

[12] ”Apache jclouds” 2013. 23 Jun. 2014 http://jclouds.apache.org/

[13] ”Apache Libcloud is a standard Python library that abstracts ...” 2011. 23 Jun. 2014 https://libcloud.apache.org/¿

[14] ”fog - The Ruby cloud services library.” 2008. 23 Jun. 2014 http://fog.io/

[15] ”pkgcloud/pkgcloud GitHub.” 2014. 23 Jun. 2014 https://github.com/pkgcloud/pkgcloud

[16] ”Cloud Portfolio Management by RightScale.” 2003. 23 Jun. 2014 http://www.rightscale.com/

[17] ”Scalr Enterprise Cloud Management Platform.” 2004. 23 Jun. 2014 http://www.scalr.com/

[18] ”OnApp.” 2008. 23 Jun. 2014 http://onapp.com/

[19] ”Cloud.net - The Global Cloud Marketplace.” 2014. 23 Jun. 2014 https://cloud.net/

[20] ”AWS — Amazon Route 53 - Domain Name Server - DNS ...” 2010. 23 Jun. 2014 http://aws.amazon.com/route53/

[21] ”AWS Elastic Load Balancing - Cloud Network Load Balancer.” 2009. 23 Jun. 2014 http://aws.amazon.com/elasticloadbalancing/

[22] ”AWS — Amazon Elastic Block Store (EBS) - Incremental ...” 2008. 23 Jun. 2014 http://aws.amazon.com/ebs/

[23] ”GSLB.me - Smart DNS Services: Authoritative, dynamic and ...” 2011. 23 Jun. 2014 http://www.gslb.me/

[24] ”GeoIP - MaxMind.” 2012. 23 Jun. 2014 https://www.maxmind.com/en/geolocation_landing

[25] ”geoNear - MongoDB Manual 2.6.3.” 2012. 23 Jun. 2014 http://docs.mongodb.org/manual/reference/command/geoNear

[26] "Percona." 2006. 23 Jun. 2014 http://www.percona.com/

[27] "OpDemand — Open source PaaS powered by Docker and ..." 2010. 23 Jun. 2014 http://opdemand.com/

[28] "CloudFlare — The web performance & security ..." 2009. 23 Jun. 2014 https://www.cloudflare.com/

[29] "Amazon Route 53 Adds Latency Based Routing." 2012. 23 Jun. 2014 https://aws.amazon.com/about-aws/whats-new/2012/03/21/amazon-route-53-adds-latency-based-routing/

[30] "Multitier architecture - Wikipedia, the free encyclopedia." 2003. 23 Jun. 2014 http://en.wikipedia.org/wiki/Multitier_architecture

[31] Streibelt, Florian et al. "Exploring EDNS-client-subnet adopters in your free time." Internet Measurement Conference 23 Oct. 2013: 305-312.

Network Telemetry Anonymization for Cloud Based Security Analysis - Best Practices

Sashank Dara* †
* Cisco Systems India Pvt Ltd,
† International Institute of Information Technology,
Bangalore, India
email: sadara@cisco.com

***Abstract*—Availability of network telemetry data aides in identifying security compromises, malicious traffic patterns, malware spread etc. There are variety of Cloud based security services available for consumers to benefit from but on an other hand there is a compelling need for ensuring privacy of sensitive fields before data is shared with any cloud provider.**

Anonymization techniques based on micro-data or macro-data have challenges in terms of attacks possible, scalability and practicality. In this paper we discuss challenges in privacy-preserving cloudification of network telemetry data. We present practical and scalable techniques for network data anonymization. These techniques ensure the privacy of the sensitive fields while retaining the ability to perform security forensics and analytics. We also provide best practices for ensuring successful data anonymization.

I. Introduction

Network telemetry data could be network flow records (NetFlow, IPFIX), web access logs, syslog events, SNMP traps etc. Such telemetry is valuable resource for identifying malicious behavior, malware spread and security forensics. Such data also could be used for network performance optimization, Quality of Service(QoS) and other research purposes too.

There are number of benefits in outsourcing such telemetry data in order to leverage from the services of Cloud providers[1],[2],[3] but ensuring the privacy of sensitive data in such telemetry is of paramount importance. Network telemetry data needs to be anonymized before it is shared.

There is wide spread paranoia among the users due to backfire of anonymization measures in publicly released datasets by AOL, Netflix. AOLs anonymization was poor [4]. PII was present in many of the queries that enabled attackers to identify the users in the data easily. In case of Netflix, there was so much publicly available information on the Internet (IMDB ratings by users) apart from the anonymized datasets released by them. These enabled researchers to try inference based attacks and break the anonymization [5].

In this paper we discuss the many challenges of both these anonymization techniques. Among both micro data anonymization based on encryption is feasible and practical in cloud based network security monitoring scenario. We propose a framework for successful privacy preserving anonymization while retaining the ability to perform security and forensic analytics.

II. Preliminaries

A. NetFlow, IPFIX

NetFlow is a technology to collect IP network traffic and perform analytics, measurements and security monitoring of the network. It is been found very useful in both network security monitoring [6] and security incident response [7]. An example packet format is given in figure 1. NetFlow allows to capture all the characteristics of the network traffic flows like source address, destination address, port, number of packets etc. It is supported by many modern networking devices.

A similar IETF standard format is IPFIX. For all practical

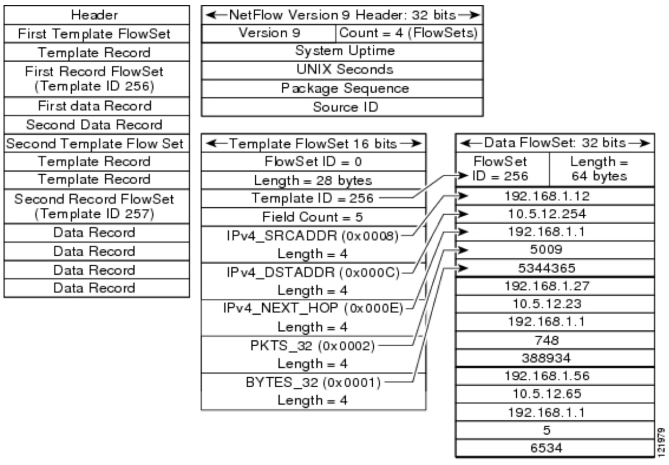


Fig. 1: Example NetFlow v9 Packet

purposes in rest of the paper, the challenges and techniques outlined for NetFlow hold good for IPFIX without loss of generality.

B. System for Internet Level Knowledge (SiLK)

SiLK [8] is an efficient network flow collection and storage infrastructure that will accept flow data from a variety of sensors. It is developed by Network Situational Awareness (NetSA) group at Carnegie Mellon University. SiLK provides a suite of efficient command-line tools for analysis. The packing sub system collects IPFIX, NetFlow v9, or NetFlow v5 and converts the data into a more space efficient format, recording the packed records into service-specific binary flat files. The

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

analysis subsystem has many utilities to perform flow statistics, top IP address counts, attempt for any scan activities in the network and many more [8].

III. Technical Goals

Cloudification of application stacks, legacy systems, databases etc. has many benefits but ensuring privacy of sensitive fields might become messier and challenging.

As a toy application, let us consider a Cloud service that performs analysis using SiLK tools on NetFlow v9 records. Privacy of sensitive fields like internal private IPv4 addresses to be ensured before consumers can share their telemetry data. In reality other telemetry data like web access logs, SNMP traps, syslog events etc. may be needed too for advanced analytics. For the sake of the discussion we restrict ourselves only for NetFlow records. There is certain amount of re-engineering to be done

We refer to such cloud application as *Data Anonymization and Analytics (DAA)*. In short, *DAA* application aides to anonymize the network telemetry data and also ensures the ability to perform analytics on anonymized data. There are two major components of *DAA* application. Refer to figure 2 for clarity.

1) The data collection and anonymization components of *DAA* reside in the *Consumer network*.
2) The analytics components and storage of results in *DAA* reside in the *Cloud Provider's* infrastructure.

We attempt to list the technical goals to be achieved for a successful cloudification of such toy application.

1) Data Anonymization: The network telemetry data should be anonymized for any traces of internal IPv4 addresses. Any related fields in the network flow data should also be anonymized. Sound anonymization techniques should be used in order to prevent the Cloud provider being able to identifying the real data.

2) Data Analytics: All the analytic operations that are possible on network telemetry data (say through SiLK command line utilities) in plain form should be also possible on the anonymized data. Such analytics should not reveal or leak any additional information to the Cloud provider.

3) Minimal Re-engineering: The components of the *DAA* application like data collection, analytics and storage should need minimum changes to be done in order to perform anonymization and analytics on the data. Such changes should not open up new attack vectors that are other wise absent.

4) Reversibility: The telemetry data that is anonymized and shared with cloud provider needs to be de-anonymized back in the consumer network. This could be to avail the results of analytics performed on the anonymized data in the cloud or for any reports generation back in the consumer network.

5) Scalability: As the networks continue to become complex, the data collectors that aggregate and transform the network telemetry to Cloud need scalable techniques for performing anonymization. *Extract-Transform-Load (ETL)* paradigm should be preferred. This paradigm aides to perform the anonymization transformations at the collectors itself before the data is uploaded to the cloud. This would aid in parallel, scalable anonymization to be done by many devices and load the telemetry into cloud independently as shown in figure 2.

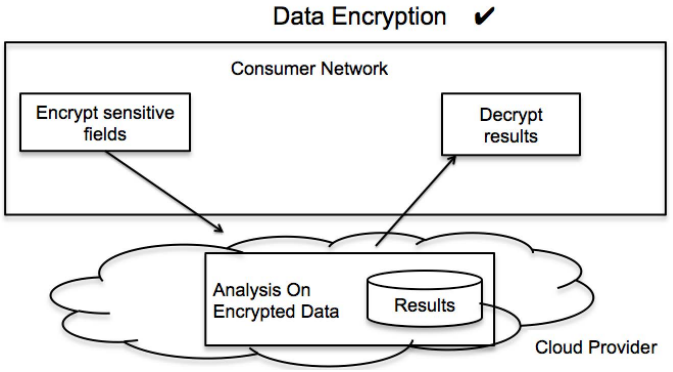


Fig. 2: Extract Transform Load

The other paradigm *Extract-Load-Transform (ELT)* will not be scalable. It assumes presence of high-end anonymization server at consumers premises before the telemetry is uploaded for performing the transformations as shown in figure 3. Conventional techniques like tokenization fall in this category.

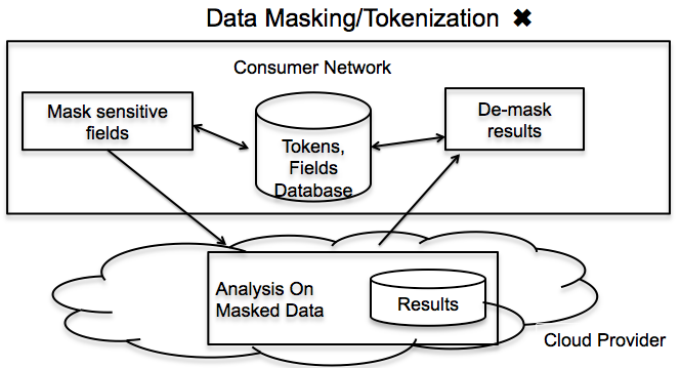


Fig. 3: Extract Load Transform

IV. Challenges

Data anonymization techniques could be broadly classified as Macro-Data based or Micro-Data based.

1) Macro data Anonymization : These techniques are based on anonymization of the data at record level, they are widely popular in data mining area. These techniques involve adding synthetic records (noise) to achieve privacy [9] [10].
2) Micro data Anonymization : These techniques work at field level (or column level). Data fields that are considered sensitive are anonymized using either encryption based techniques or tokenization/masking techniques. Non-sensitive data fields in the records are left as is.

A. Macro data Anonymization

These techniques are based on perturbation of data records [11]. There are challenges with these techniques

1) They need high noise levels to achieve good epsilon (privacy) values this means we add more dummy packets to real packets to obfuscate stuff. Which is not scalable too.
2) A practical challenge is adding quality "synthetic records" that mimic the data. As most of the security parameters of academic formalizations like *k-anonymity, l-diversity and I-closeness* to achieve good privacy levels depend on them. This further pushes the complexity in practice.
3) The data is not really anonymized its only perturbed/obfuscated. This is dangerous as real data is as is. Imagine user names retained (for example in web logs) in the perturbed anonymized data, attacker simply does a search on Facebook or LinkedIn with good probability to make out of the name is synthetic or real.
4) Limited number of statistical operations are possible say histograms, count, aggregates, averages, mean, mode and such like. These may not be useful in scenarios like network security forensics.

B. Micro data anonymization

Privacy preserving efforts involves identification of sensitive fields in the system and re-engineering such data fields in order to encrypt them. Variety of techniques exists for field level anonymization like data removal, randomization, generalization, truncation, bijective mappings, encryption, hashing etc. A good survey could be found here [12].

These techniques anonymize sensitive data fields (or columns) in the record using cryptographic techniques like encryption or conventional techniques like tokenization or masking. There are many challenges in these techniques too which in later sections we try to address.

1) Preserving Lengths and Formats: Sensitive data fields in network telemetry are often part of well defined rigid formats like NetFlow, IPFIX. For example, a data field might be well-defined for accommodating IPv4 address of size 32 bits with number of validations throughout the application stack (and/or database) to ensure the input data is in fact IPv4 address.

In order to preserve the privacy of this field using traditional AES-128 scheme it has to be re-engineered to accommodate 128 bits of memory instead of 32 bits. Also any validations that identifies the type of the data (dotted notation of IPv4 address) needs to be removed in order to accommodate a random cipher string of encrypted IPv4 address.

While the length expansion of the field needs more storage (and/or bandwidth), removal of validations might open up serious security vulnerabilities in the form of injection attacks [13].

Techniques like *TCPDPriv* [14] are not parallel in nature, they require all the telemetry data to be aggregated at one common server before the data could be anonymized. Otherwise IPv4 addresses anonymized by different collectors would not be deterministic and we lose the ability to perform analytics. This approach falls in the category of *Extract-Load-Transform* paradigm figure 3. As stated in the previous section this does not meet our technical goals.

The prefix-preserving encryption technique *Crypto-PAN*[15] has been proposed to encrypt the IPv4 addresses. These techniques ensure that subnet of the anonymized address match too. They are considered weak and prone to attacks [16]

For these reasons, good encryption techniques are to be chosen for data anonymization that preserve the length and formats of the data fields too.

2) Inferential Attacks: These techniques don't attack the cryptography primitives or implementations but based on inferences derived from rest of the data fields that are not anonymized [17] *Fingerprinting/signature attacks* and *Injection attacks* are popular ones on anonymization of NetFlow records and logs.. These attacks are made possible in two steps

1) The attacker monitors customers network (for finger printing hosts) or attacker can inject packets of his choice in the customer's network and nothing beyond that.
2) The attacker then reads the net flow records post anonymization to infer the fields that are anon'ed based on finger prints of non anon'ed fields or characteristics of packets he injected.

Note that same challenges are applicable while ensuring privacy of any sensitive fields like *MAC addresses, Email addresses, Usernames, Account numbers, Serial Numbers, SSN, Credit Card numbers* etc. in other forms of network telemetry like web logs, syslog events, SNMP traps etc.

V. Methodology

Of all the data anonymization techniques we recommend usage of encryption based techniques for below reasons

1) Using good encryption algorithms can ensure soundness of the security in anonymizing. These encryption techniques should be standard algorithms like AES or techniques that are dependent on AES.
2) Clients in Consumer network can De-anonymize (reversibility) data fields is possible, in presence of a secret key. This is needed for availing the results of security forensics and analytics performed in the cloud.
3) Does not involve additional storage needed for substitution or mappings based techniques like tokenization.

But as we have seen in section IV, there are number of challenges in using micro-data anonymization techniques. We address the challenges of length and formats of data fields using recent developments in cryptographic techniques. We then address the challenges of inferential attacks through classification and enlisting of each data field in the record and recommending a particular technique to anonymize the same using *DAA Matrix*.

A. Format Preserving Encryption

Format Preserving Encryption(FPE) has been studied rigorously [18], [19]. An introductory white paper is also available from Voltage Inc. [20]. FFX modes of operation that aide to

preserving lengths and formats of the data have been submitted for standardization process too [21].

Any FPE technique is basically a combination of *length preserving* block cipher and *ranking functions*. Ranking functions, as the researchers say[18], is a *folklore* approach to preserve the formats of the data types.

For example, an IPv4 address could be ranked as a 32 bits integer. The ranked integer is then encrypted using a block cipher like FNR to result in another 32 bits integer. The resultant cipher text is converted back to dotted notation of IPv4 address in order to preserve the format.

FNR is a recently proposed length preserving block cipher that offers better security bounds than FFX modes of operation [22] [1]. It is free of any intellectual property restrictions and available under open source license [23]. It could be used when the size of sensitive fields is 32 to 128 bits. For example encrypting IPv4 (32), MAC (48), Port (16), IPv6 (128) etc.

FNR encryption scheme, like other FFX modes, is tweakable in nature. A tweak is additional input to the block cipher along with the plain text and key. The tweak is more like cryptographic salt and could be made public. It offers additional randomness to the cipher text. Since the FNR is intended to use for smaller data types, a tweak is designed inbuilt .

Similar approach could be pursued for preserving lengths and formats of other data types like IPv6, MAC addresses etc. Note that since IPv6 address is 128 bits in length traditional AES-128 itself could be used to both encrypt and preserve the IPv6 address format.

B. DAA Matrices

While anonymizing fields of particular data format, the purpose of each data field should be thoroughly analyzed. The type of the field should be determined first and then appropriate anonymization technique should be applied. The anonymizing technique for that particular field further depends on such analysis.

The workflow is as described in the figure 4.

The below classification is needed in order to ensure the *DAA* process does not leak any additional information to an adversary.

There are predominantly three different types of fields and as we shall discuss we need different techniques of anonymization for each such type as shown in figure 5

1) Direct: These are data fields that are considered sensitive and directly needed for analytics. Consistent anonymization of the values of these fields may be required for correlation across multiple homogeneous sources (different telemetry collectors, devices) and/or multiple heterogeneous sources (different logs like Web Server Logs, NetFlow records etc.) Anonymization of these fields would also require de-anonymization upon providing proper authentication material like keys.

[1]FNR paper is accepted in SPACE-2014 as of this writing

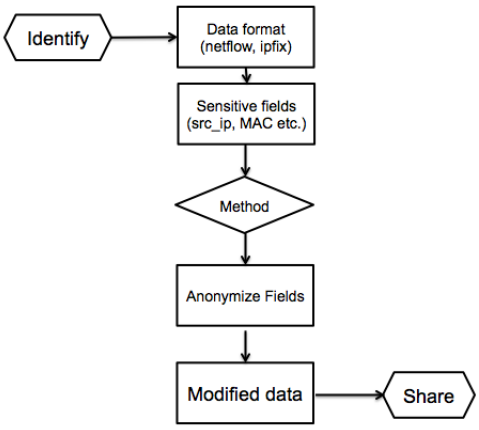


Fig. 4: DAA Work Flow

	Direct	Indirect	Public	Technique
Field1	Yes	No	No	Deterministic Encryption
Field2	No	Yes	No	Randomized Encryption
Field3	No	No	X	Removal (Nullify)
Field4	Yes	No	Yes	Plain (As is)
Field5	Yes	No	Yes	Plain (Modify/Truncate)

Fig. 5: DAA Matrix

a) Deterministic Encryption: In order to run analytics, consistent anonymization of few data variables is needed. For example a SRC_IP in NetFlow record may need to be compared with IP address in a web access log. Another example an analyst may need to find out an SRC_IP address that appeared the most in the anonymized NetFlow records, for achieving this functionality the SRC_IP address should be deterministically encrypted no matter which collector anonymizes the flow records. So for this reason, irrespective of the anonymizing source the encryption of the data field SRC_IP should be deterministic in nature.

Consistent anonymization of a particular field can be achieved by keeping the tweak and secret-key of FNR encryption scheme same. Traditional AES-128 in ECB mode can be used if length preservation is not a constraint.

2) Indirect: These are data fields that are sensitive and not directly needed for analytics but needed to be part of final results or reports. Randomized anonymization of the values of these fields is required so that we dont leak any information for attackers to infer from anonymized data. For example FLOW_SET_ID may not be needed for running analytics (and/or correlation) but may be needed for forensics or results. Consistently encrypting this parameter may leak additional information to attacker that few NetFlow records can be grouped together. Anonymization of these fields would also require de-anonymization upon providing proper authentication material like keys.

a) Randomized Encryption: Such data fields should be encrypted using randomized techniques (like different IV's or tweaks etc.). This means the collector anonymizing the records should have specific IVs, tweaks or salts used. Failing to use randomized encryption of the data fields that are not

directly needed for analytics may result in inference attacks. Randomized Encryption of a particular field can be achieved by changing the tweak of FNR encryption scheme. Care should be taken though that the tweak is available/known to the decryptor.

3) Public: These are data fields that are not sensitive and public nature. They can be left as is or with minor sanitization.

a) No Anonymization (or Plain): For example, DST_IP 8.8.8.8 can be left as is as the server is public. A web log file containing GET URL *www.linkedin.comprofile?usr=john_zyx* may need to be sanitized by removing the user Id in the URL.

4) Removal (or Nullifying): Data fields not needed either for analytics or reversal should be nullified or removed completely from the data records. For example protocol version. Failing to do so may leak unwanted information for attackers to try inference attacks.

C. Examples

Few example *DAA Matrices* are given just for illustration purposes may vary with actual use case. An example NetFlow v9 packet format [24] is in figure 1 and its corresponding DAA Matrix is given in figure 6.

	Direct	Indirect	Public	Technique
FlowSetID	No	Yes	No	Randomized Encryption
Length	No	Yes	No	Randomized Encryption
TemplateID	No	Yes	No	Randomized Encryption
FieldCount	Yes	No	No	Deterministic Encryption
IPv4_SRCADDR	Yes	No	No	Deterministic Encryption
IPv4_DSTADDR	Yes	No	Yes	Plain
IPv4_NextHOP	No	No	No	Remove (Nullify)
PKTS	No	No	No	Remove (Nullify)
BYTES	No	No	No	Remove (Nullify)

Fig. 6: Example DAA Matrix for NetFlow v9 Data Flow

D. Advanced Analytics

Current use cases for doing analytics on anonymized data need exact matching of anonymized data fields across multiple data sources (like telemetry devices, different log files etc.) Advanced analytics like prefix, suffix, and substring matches on the anonymized data are possible too [25].

VI. Threat Model

There are predominantly two classes of attacks on anonymization techniques

1) **Type-1** Attacks on the underlying cryptographic implementations, key management systems, poor entropy of keys, brute force attacks. These attacks are common to any techniques that are based on encryption.
2) **Type-2** Attacks due to inferring additional knowledge from anonymized datasets. This could be due to fingerprinting the available network resources or even injecting packets of one's choice in the network to correlate it with anonymized data.

It is important to understand the attackers capabilities and information about the system they possess in order to safe guard it. Both internal and external attacks need to be accounted for while threat modeling the entire system that performs *DAA*.

The attacks could happen at various places as shown in the figure 7. Various attacks possible at each of these parts of the system are discussed in detail below.

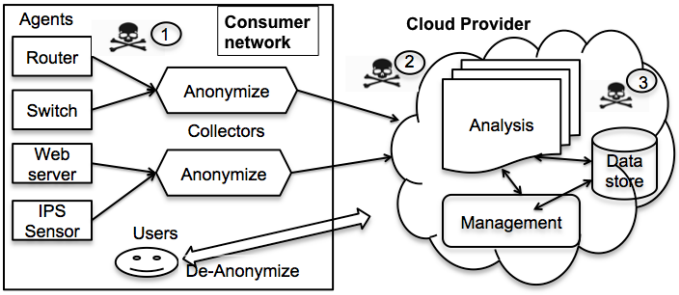


Fig. 7: Threat Model

A. Consumers Network

Within the Consumers network the telemetry devices that perform the anonymization and also the client machines where users de-anonymize the results to view the reports of security analysis are prone to variety of attacks. These attacks could be performed by either internal or external adversaries.

1) Telemetry devices: The network telemetry could be collected, aggregated and anonymized in various devices like routers, switches, sensors or dedicated log collectors too. These typically reside in the consumers network.

2) Client Machines: The client machines could be work stations, browsers, laptops or mobile devices where the users would like to de-anonymize the security analysis results and view them.

B. Data-in-Transit

Data-in-Transit from Consumers network to Cloud provider.

C. Cloud Provider

Cloud Providers infrastructure where the analytics are performed or where the anonymized data is storage (i.e data-at-rest).

VII. Best Practices

Measures for mitigating various attacks described are discussed in detail here.

1) Mitigating Type-1:

1) Measures to mitigate these types of attacks include expert reviews of the specifications, implementations of the cryptographic techniques developed by both internal and external security researchers. The proposed FNR mode is based on proven theoretical constructions and relies on strong AES encryption internally.

2) The encryption done on smaller fields like 32 bits of IPv4 address can be easily broken if the attacker could get hold of one pair of plain text and corresponding cipher text. Brute forcing all possible combinations of bits can thus reveal the key (assuming the key is small) and collapse the security of entire system. In order to mitigate these an additional parameter tweak is designed into FNR mode. This forces the attacker to get more such plain text and cipher text pairs in order to break into the system rather than just one.
3) It is important not to hard code secret keys, IVs, tweaks needed by the encryption techniques in the code, scripts etc. that perform *DAA*. It is always a good practice to provision such keying material using standard key management techniques prevalent in the organization.
4) In the absence of a key management service at the Consumers network, strong password based key derivate functions like PBKDF2 should be used.
5) It is always important to generate the secret keys from a good entropy source (than */dev/random* which is shown to be provide poor entropy)
6) Preserving the formats of the anonymized data would help in mitigating SQL Injection and other attacks based on input validations.

2) Mitigating Type-2: Inferential attacks could be mitigated by

1) Thoroughly inspect the need for each data field in the system and form the *DAA Matrices* accordingly.
2) Randomized encryption should be used to extent possible to mitigate identification of patterns in the anonymized data.
3) Deterministic encryption is considered weaker and should be used frugally.
4) Data fields that are not used should be nullified.
5) Sound network security practices within the customers network to prevent adversaries from performing any *Finger printing* or *Packet Injection* in order to draw inferences from the anonymized data.
6) Sound network security practices at cloud providers network to mitigate unauthorized access to the anonymized data.
7) Multiple layers of defense in depth, access control to the devices performing *DAA* and even the ones performing analytics.
8) Transferring the anonymized data over an SSL connection might mitigates attacks based on eaves dropping when the data is in transit.

VIII. Limitations

Key compromises could be costly. Once data is in anonymized and exported to cloud, any key compromises would need the entire data to be downloaded into the Enterprise premises and re-anonymized with new keys and uploaded back into the cloud. This may not be feasible if the Enterprises dont have provisions to download their entire data. For this reason stronger key management practices should be in place to mitigate key compromises.

In situations where the key derivation is done from a user-supplied password, change of password would be costly for above said reasons.

On a side note, if the anonymized data is short lived in the cloud, for example if parts of anonymized telemetry uploaded is used for analytics and later discarded in shorter cycles, this limitation may not be serious. But if the anonymized data is archival or for long term purposes then the data size might grow hugely and downloading it back into enterprise networks (in case of key compromises) might be costlier for any re-anonymization with newer keys.

IX. Conclusions

Data anonymization is a tough problem. It is very tight walk, especially because the anonymized data should still be usable for performing any analytics. In this paper we presented methods, techniques and best practices to perform a successful network telemetry anonymization in order to benefit from any Cloud based network security monitoring and forensic applications.

Acknowledgments

I would like to thank Cisco Systems for supporting this work. I would like to thank Dr David McGrew, Scott Fluhrer, Anthony Grieco, Dr. Madhav Marathe and Dr. Martin Rehak for many insightful suggestions and improvements in this work. I would also like to acknowledge and thank my immediate management chain Sharad Chandra and Anil Nair for supporting me while carrying out this work. I would also like to thank CISCO for making the FNR encryption scheme available under open source license. I would like to thank my colleague Bhanu Prakash Gopularam for proof reading and corrections.

I would like to thank my research advisor Dr V.N. Muralidhara at IIIT-Bangalore for all the insightful discussions and brainstorming on Cryptography. I would also like to thank *crypto.stackexchange* community for having many insightful discussions around this topic.

References

[1] http://www.lancope.com/.
[2] http://www.logicmonitor.com/about/news/news/logicmonitor-introduces-free-netflow -network-monitoring-conjunction-nation-2013-sponsorship/.
[3] S. Wilkins, "Top 4 Cloud Network Monitoring Tools Compared," http://www.tomsitpro.com/articles/cloud-network-monitoring-tools,2-707.html, 2014, [Online; accessed March 20, 2014].
[4] S. HANSELL, "TAOL removes search data on vast group of web users." http://query.nytimes.com/gst/fullpage.html ?res=9504E5D81E3FF93BA3575BC0A9609C8B63, 2006, [Online; accessed August 8, 2006].
[5] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Security and Privacy, 2008. SP 2008. IEEE Symposium on*. IEEE, 2008, pp. 111–125.
[6] C. Mike, "Netflow security monitoring for dummies." Wiley.
[7] ——, "Incident response with netflow for dummies." Wiley.
[8] T. Shimeall, S. Faber, M. DeShon, and A. Kompanek, "Using silk for network traffic analysis," tech. rep., CERT Network Situational Awareness Group, Tech. Rep., 2010.
[9] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[10] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, p. 3, 2007.

[11] G. Cormode and D. Srivastava, "Anonymized data: generation, models, usage," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. ACM, 2009, pp. 1015–1018.

[12] T. Farah, "Algorithms and tools for anonymization of the internet traffic," Ph.D. dissertation, SIMON FRASER UNIVERSITY, 2013.

[13] D. A. Kindy and A.-S. K. Pathan, "A detailed survey on various aspects of sql injection in web applications: Vulnerabilities, innovative attacks, and remedies," *arXiv preprint arXiv:1203.3324*, 2012.

[14] G. Minshall, "Tcpdpriv," 1997.

[15] A. Slagell, J. Wang, and W. Yurcik, "Network log anonymization: Application of crypto-pan to cisco netflows," in *Proceedings of the Workshop on Secure Knowledge Management 2004*, 2004.

[16] T. Brekne, A. Årnes, and A. Øslebø, "Anonymization of ip traffic monitoring data: Attacks on two prefix-preserving anonymization schemes and some proposed remedies," in *Privacy Enhancing Technologies*. Springer, 2006, pp. 179–196.

[17] R. Pang, M. Allman, V. Paxson, and J. Lee, "The devil and packet trace anonymization," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 1, pp. 29–38, 2006.

[18] M. Bellare, T. Ristenpart, P. Rogaway, and T. Stegers, "Format-preserving encryption," in *Selected Areas in Cryptography*. Springer, 2009, pp. 295–312.

[19] P. Rogaway and D. Tweet, "Format-preserving encryption," 2010.

[20] T. Spies, "Format preserving encryption," *Unpublished white paper, www. voltage. com Database and Network Journal (December 2008), Format preserving encryption: www. voltage. com*, 2008.

[21] M. Bellare, P. Rogaway, and T. Spies, "The ffx mode of operation for format-preserving encryption (draft 1.1). february, 2010," *Manuscript (standards proposal) submitted to NIST*.

[22] S. Dara and S. Fluhrer, "Fnr : Arbitrary length small domain block cipher proposal," in *Security, Privacy, and Applied Cryptography Engineering*. Springer, 2014.

[23] https://github.com/sashank/libfnr.

[24] http://www.cisco.com/c/en/us/td/docs/ios/12_2sr/12_2srb/feature/guide/nfv6xsrb.html.

[25] S. Dara, "Advanced searchable encryption over well defined strings," in *Emerging Research in Computing, Information, Communication and Applications*. Elsevier, 2014.

Outsourcing Resource-Intensive Tasks from Mobile Apps to Clouds: Android and Aneka Integration

Tiago Justino and Rajkumar Buyya
Cloud Computing and Distributed Systems (CLOUDS) Laboratory
Department of Computing and Information Systems
The University of Melbourne, Australia
{tiago.vieira, rbuyya}@unimelb.edu.au

***Abstract*—Mobile Cloud Computing enables augmenting mobile device capabilities and increasing battery lifetime through the extension of cloud services and resources, resulting in an enhanced user experience. However, the development of a mobile cloud application is challenging because it involves dealing with different cloud providers and mobile platforms. To tackle the above issues, a mobile cloud architecture is proposed to asynchronously delegate resource-intensive mobile tasks in order to alleviate the mobile device load and, consequently, extend the battery life. We demonstrate this capability by developing an interface that supports the delegation of heavy tasks from mobile apps running under the Android mobile platform to a cloud computing environment managed by the Aneka Cloud Application Platform. The Aneka Mobile Client Library encapsulates the processes of communicating to cloud is provided, thus, the effort and complexity of developing a mobile cloud application is decreased. Two different resource-intensive mobile application are presented in order to show the library effectiveness. A performance evaluation is conducted showing the feasibility of architecture through the reduction of application execution time and extension of mobile device battery life.**

***Keywords*—Mobile Cloud Computing, Task Delegation, Mobile Apps, Android, Aneka, Software Engineering.**

I. Introduction

The International Telecommunication Union (ITU) expects the number of mobile phone accounts to exceed the global population in 2014[1]. This growth is observed throughout all the Smart Mobile Devices (SMDs) domain, such as: Personal Digital Assistants (PDAs), smart phones, and tablets. In recent years, the advance in semiconductor technology enabled the design of mobile devices increasingly powerful and compact [1]. Although the technology has advanced, the miniature and mobile nature of these devices impose intrinsic limitations on CPU, memory, and battery lifetime [2], [3].

This technological advancement enabled the execution of resource-intensive mobile applications, such as voice recognition, image processing, optical character recognizers, and online games. However, SMDs rely on finite energy source and they are resource-poor compared to stationary machines such as desktops and servers [4]. Nevertheless, users presume to execute resource-intensive applications on their SMDs with the same quality expectations (performance and reliability) [3].

In this context, Mobile Cloud Computing (MCC) enables augmenting SMD capabilities through extension of cloud services and computational resources to SMD on demand [3]. The augmentation of capabilities includes screen, battery life, storage, and application processing (CPU, memory) [5]. Thereby, storage and processing are increased and battery lifetime is extended, enhancing the user experience. Furthermore, this solution inherits characteristics intrinsic to cloud environments like, pay-as-you-go model, elasticity, illusion of infinite resources, and task parallelization [6].

However, the development of a mobile application that requires accessing distributed hybrid clouds is challenging because it involves dealing with different Web APIs from different cloud providers (e.g., Amazon, Microsoft Azure) and different mobile platforms (e.g., Android, iOS, Windows Phone). Moreover, porting these API to SMDs is a difficult task due to compiler limitations, additional dependencies, and code incompatibility reasons [6]. To tackle the above issues, a mobile cloud architecture is proposed to delegate in asynchronous manner resource-intensive mobile tasks in order to alleviate the mobile device load and, consequently, extend the battery life.

We demonstrate this capability by developing an interface that supports the delegation of heavy computing tasks from mobile apps running under the Android mobile platform to a cloud computing environment managed by the Aneka Cloud Application Platform. However, our proposed model for integration of Android and Aneka platforms can be easily applied to other mobile platforms such as iOS and Windows Phone.

The **main contribution** of this paper is the Aneka Mobile Client Library for Android platform that encapsulates the processes of connecting to cloud, serializing and deserializing messages, sending messages, and collecting their responses. Thus, the effort and complexity of developing a mobile cloud application is decreased. In addiction, the library was designed to leverage the Aneka Cloud Application Platform, which provides transparent resource provisioning and job scheduling services and encapsulates different cloud providers Web APIs. The user has no concern in allocating or deallocating virtual machines or distributing the jobs among the resources.

This paper also presents two different resource-intensive mobile applications (ray tracing image generation and Mandelbrot set generation) in order to show the Aneka Mobile Client Library effectiveness. A performance evaluation is conducted with objective of comparing the applications execution in the mobile device and delegating to the cloud in view of metrics battery consumption and processing time.

[1] http://www.siliconindia.com/magazine_articles/World_to_have_more_cell_phone_accounts_than_people_by_2014-DASD767476836.html

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

The rest of this paper is organized as follow: Section II analyses related works, highlighting similarities and differences for this proposal. Section III describes the architectures and software artifacts, and its deployment. Section IV describes the Aneka Mobile Client Library development and shows its operations. Section V presents two different resource-intensive mobile applications developed using the proposed approach in order to evaluate it. Finally, Section VI concludes the paper and highlights future research directions.

II. Background and Related Work

In order to augment the processing power of mobile devices, two main approaches have emerged [6]: offloading and delegation. In offloading, applications are partitioned into components, which are analyzed in order to determine whether they should be migrated to the cloud. This process can occur at development time or at runtime. These components can be different entities depending on the granularity level, such as method, class, module, application partition, entire application or image [3]. Usually, the analysis considers two variables for deciding about migration: the amount of computation and the amount of data to communicate. A component is migrated if intensive computation and little communication are needed [1]. The finer the granularity, the more intensive is the synchronization mechanism between mobile device and cloud. The more abstract the granularity, the more traffic intensive is the communication [3].

Shiraz et al. [3] describes and compares 17 different works using offloading, considering several aspects such as partitioning approach and migration granularity. Among them we highlight the two following works. Hung et al. [7] propose an Android framework based on VM migration for application offloading. The framework installs an application on the mobile phone for managing the offloading process. This application encapsulates other running applications in VMs to migrate and execute on cloud servers. This approach requires a large amount of bandwidth and processing for ensuring consistency between the mobile phone and the cloud server. Cuervo et al. [8] propose Mobile Assistance Using Infrastructure (MAUI), which adopts a dynamic (runtime) partitioning approach at a method level with focus on energy saving for the mobile device. MAUI leads to extra overhead on mobile devices to analyse the amount of computation needed for each method and to migrate and reintegrate this computation.

Delegation utilizes Service Oriented Computing (SOC) in order to migrate the execution of resource-intensive tasks, e.g. prime verification or matrix multiplication, to cloud. Mobile tasks are delegated by invoking a cloud service [6] and no code is migrated to cloud, what makes this approach more lightweight than offloading. On the other hand, as delegation has dependency on the cloud service to execute part of the computation, it requires always available network connectivity.

In the offloading approach, the cloud utilization is limited to executing applications that can run on mobile devices. On the other hand, delegation approach allows the utilization of resources unavailable in the mobile device platform, but that are available in the platform running on cloud. Section V discusses an example of this advantage, where the software for image rendering POV-Ray[2], available only for the Windows platform, is used from an Android application. Furthermore, the delegation approach allows the use of other characteristics intrinsic to cloud environments, such as dynamic resource provisioning, elasticity, illusion of infinite resource, and task parallelization [6], as presented in this work.

Abolfazli et al. [2] implement mobile augmentation by utilizing delegation and analyses how the number of hops impacts the execution time in a service call. They prefer SOC instead of offloading in order to avoid the overhead of identifying, partitioning and transferring large amounts of data from mobile to cloud. They do not leverage cloud aspects like dynamic provisioning or parallel tasks execution.

Flores and Srirama [6] propose a Mobile Cloud Middleware (MCM) to work as an intermediary between the mobile phone and the cloud in order to manage the asynchronous delegation of mobile tasks to cloud resources. MCM abstracts the API and manages different cloud providers, as well as allows the development of customized services based on service composition. The mobile task computation happens on the cloud providers and a connection between MCM and the providers is kept during all the execution. After the computation is finished, MCM stores the results in the transactional space and sends a message to the mobile application via push notification, signalling the end of execution.

Although MCM communicates with the cloud resources, the mobile application is responsible for managing the information about the services and the cloud providers, which increases the mobile application complexity and reduces flexibility. If any piece of this information changes, the mobile application needs to be updated. In addition, MCM requires services to be previously developed and deployed. In this work, the Aneka cloud provides a dynamic resource provisioning allowing scale up and down according to application needs, which is not observed in MCM.

III. Architecture

This section describes the elements that compose the proposed solution architecture and how these elements relate as shown in Figure 1. The deployment of the architecture is presented in Figure 2. The two main structures in the architecture are: (i) Aneka service, which provides via web interface a runtime environment for tasks execution; and (ii) mobile client, which allows mobile applications to send resource-intensive tasks to execute on cloud and alleviate the mobile device load.

The architecture was designed in order to facilitate the development and deployment of mobile cloud applications. Firstly, by using the *Aneka Mobile Client Library*, all the complexity of communicating to Aneka cloud and submitting jobs is taken care for the developer. Secondly, by using the Aneka cloud, which provides transparently the resource provisioning and job scheduling services, the user has no concern in allocating or deallocating virtual machines or distributing the jobs among the resources.

[2]http://www.povray.org/

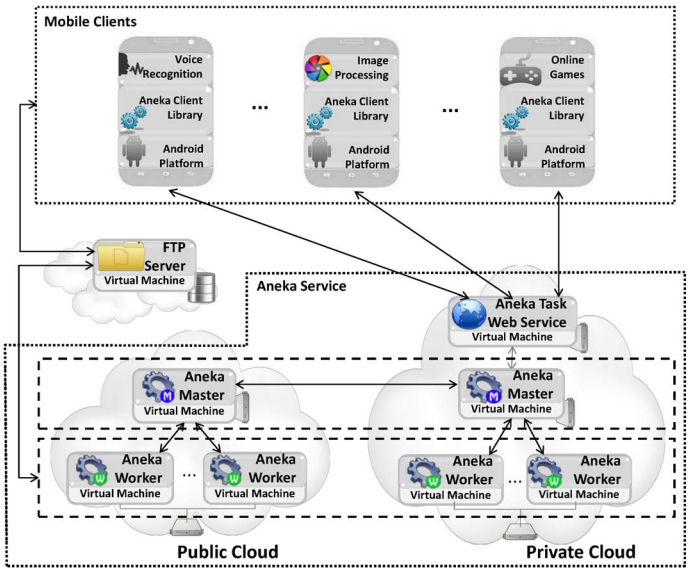


Fig. 1. Architecture showing how mobile devices interact with an Aneka cloud through the *Aneka Task Web Service* in order to outsource resource-intensive tasks.

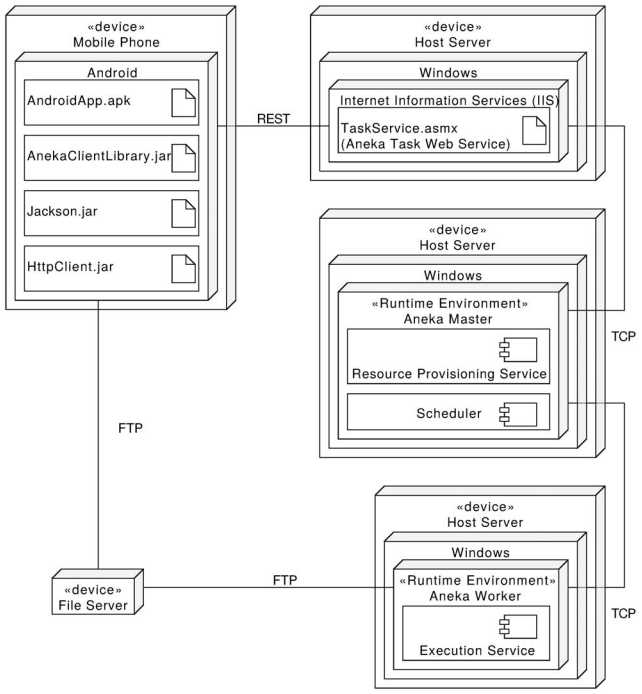


Fig. 2. Deployment Diagram showing how the different artifacts can be deployed and how they communicate.

A. Aneka Service

Aneka [9] is a PaaS solution for developing cloud applications that can be deployed on both public and private clouds. It provides a runtime environment as well as an Application Programming Interfaces (APIs) to build .NET applications leveraging the parallel power of an Aneka cloud. By using these APIs, developers can implement and deploy applications that automatically scale on demand, following different programming models such as Bag of Tasks (BoT), Thread, and MapReduce.

An Aneka cloud is defined as a collection of physical resources (desktops, servers) and/or virtualized resources (virtual machines) connected though a network, each of these machines running an instance of Aneka Container. This container represents the basic deployment unit of Aneka and it provides a runtime environment composed of *Aneka Master* and *Aneka Workers* to execute the distributed applications.

Aneka is designed following a Service Oriented Architecture (SOA), which makes it customizable and extensible, allowing developers to create new services that replace the default ones or that add new functionalities to Aneka. The default installation provides services such as dynamic resource provisioning, resource reservation, persistence, storage, security and performance monitoring [9].

The *Aneka Master* hosts the dynamic resource provisioning service, which is responsible for dynamically acquiring and integrating new *Aneka Workers* into the Aneka cloud, allowing it to elastically scale up and down to satisfy the applications needs. It also hosts the *Scheduling Service*, which is responsible for dispatching the collection of application jobs to the *Aneka Workers*, as shown in Figure 2. Thereby, when a mobile cloud application is executed, its jobs are submitted to the *Aneka Master*, via *Aneka Task Web Service*. Each of these jobs is moved to *Aneka Workers* and processed by the *Execution Service*, which is the runtime environment in charge of retrieving all the files required for execution, monitoring job execution, and collecting results.

The *Aneka Task Web Service* enables applications developed in any language to send mobile jobs to run on Aneka cloud [10]. This service exposes, via Representational State Transfer (REST), the main functionalities such as authenticate user, create application, submit jobs, and query information (see Figure 2). REST was preferred to SOAP because the message serialization process is faster [11], configuring an advantage to *Aneka Task Web Service*.

B. Mobile Client

The mobile client side of the architecture contains Mobile Cloud Applications, *Aneka Mobile Client Library*, and *Android Platform* layers, as presented in Figure 1. Mobile cloud applications, such as voice recognition, image processing, optical character recognizers, and online games, benefit from cloud computing resources to execute resource-intensive tasks.

Depending on the mobile application characteristics, input files may need to be uploaded to the cloud. For instance, an audio file needs to be uploaded for a voice recognition application or, a image file for an image filtering application. In order to keep a light implementation, the web service interface was designed not to allow user to upload or download files through it. An easy way to transfer files to use in Aneka cloud is through a storage service. As shown in Figure 2, the FTP is used to communicate with the *File Server*.

Once the mobile application developer identifies a resource-intensive task, they can instantiate the *Android Client Library* in order to consume the *Aneka Task Web Service*. The *Android Client Library* hides the complexity of delegating tasks to Aneka, executing steps, such as connecting to the

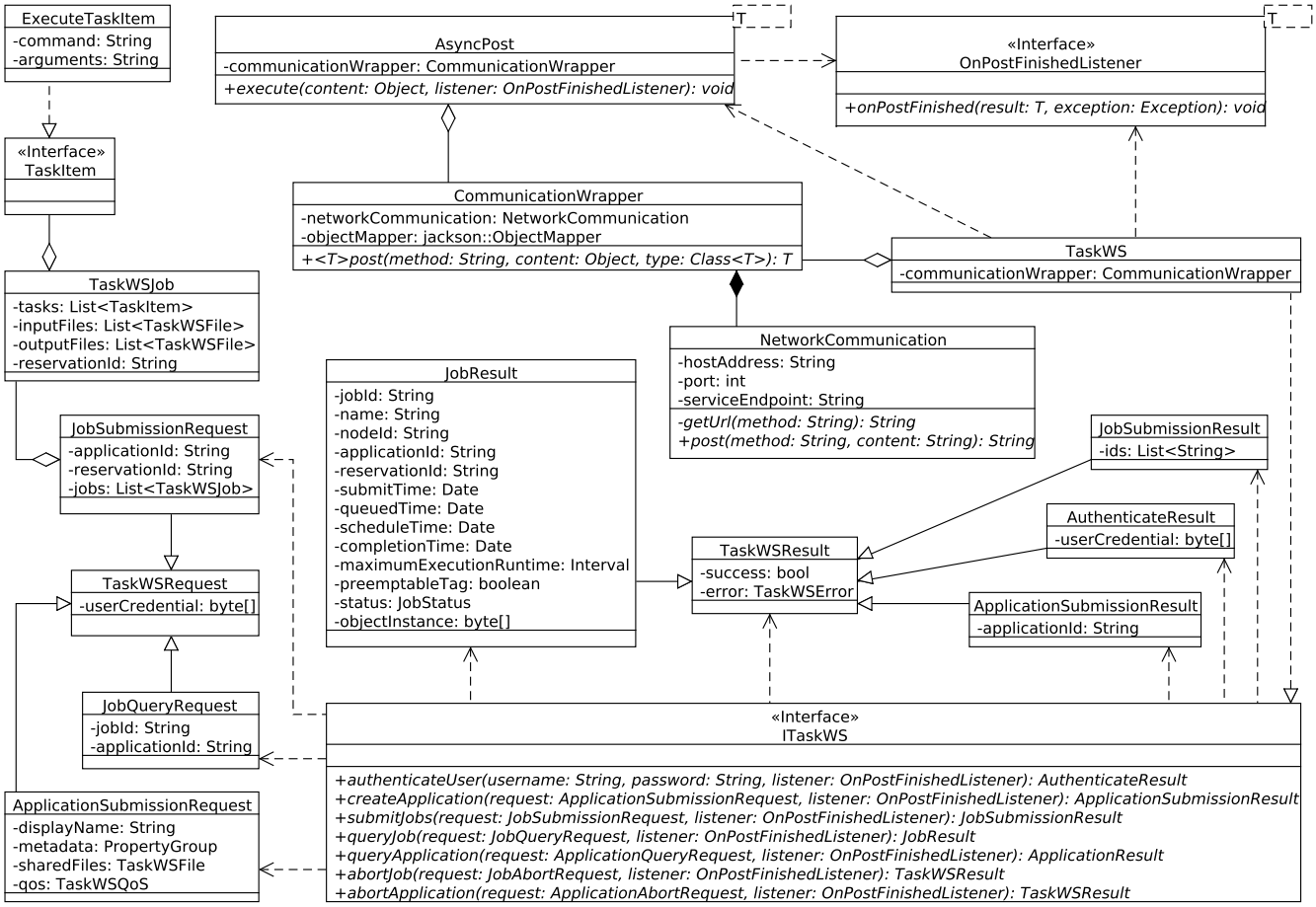


Fig. 3. Class Diagram showing the main classes and interfaces, such as *TaskWS* and *ITaskWS*, and how they interact.

server, serializing and encapsulating objects into requests, sending messages, and collecting their responses. Thus, the mobile cloud application development is simplified since the library can be easily reused by new applications.

The serialization and deserialization process of messages exchanged between the *Android Client Library* and the *Aneka Task Web Service* is provided by the third-party library called Jackson[3]. The messages are serialized to JSON format and sent to *Aneka Task Web Service* through the Http-Client class, provided by Apache[4]. The JSON format simplifies the communication between different platforms. Additionally, it is more lightweight in comparison with XML [11].

The *Android Client Library* invokes the Aneka service in an asynchronous manner since the resource-intensive task requires time to process and keeping the connection entails mobile device battery consumption. Also, these devices are prone to connection loss due to their mobility characteristic, which makes the synchronous communication an unwise choice. So, in the event of connection loss during a synchronous task execution the information gets lost and the message needs to be retransmitted, entailing consumption of time, mobile device battery, and cloud resources.

[3]http://jackson.codehaus.org/

[4]http://hc.apache.org/httpcomponents-client-ga/

The *Android Client Library* was developed in Java in order to execute over the *Android Platform*. However, the library could be developed for iOS or Windows Phone, moreover, the architecture does not depend on the mobile phone platform.

IV. Design and Implementation

This section details the design and implementation of the Android Client Library proposed in this work. This library cover all 7 operations provided by the *Aneka Task Web Service*, which are: user authentication, application creation, application query, application abortion, job submission, job query, and job abortion.

Figure 3 shows the main library classes and interfaces. The *ITaskWS* interface holds the *Aneka Task Web Service* methods. This interface is implemented by *TaskWS* class. *TaskWS* is responsible for creating and storing a *CommWrapper* instance. *CommWrapper* has only one method, called *post*, implemented using Java Generics. This method receives as parameters a string containing the web service operation to be executed, the content to be sent to the web service, and return type for which the response will be converted.

TaskWS also instantiate a *AsyncPost* object every time it needs to execute an asynchronous operation with the web service. *TaskWS* transfer its *CommWrapper* instance to *AsyncPost*

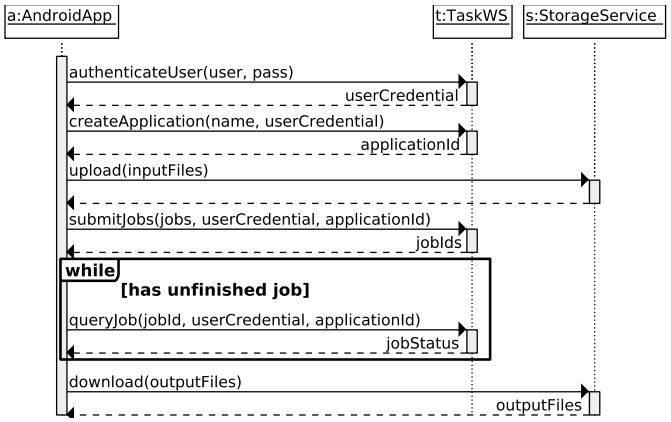


Fig. 4. Basic lifecycle for a mobile application using Aneka.

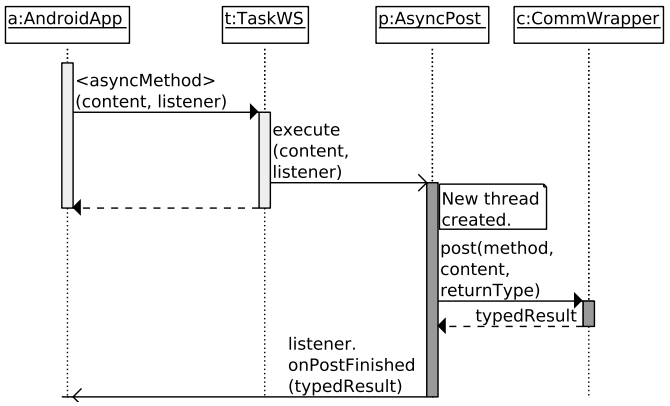


Fig. 5. Sequence diagram for asynchronous methods.

through the constructor. *AsyncPost* implements the *AsyncTask* interface from Android API, which is responsible for creating thread for executing the *CommWrapper post* method. *CommWrapper* uses the *NetComm* class, which is responsible for generating the HTTP message, assembling the URL for HTTP call and sending the message to the web service through the apache *HttpClient* class. *TaskWS* needs also to give *AsyncPost* an *OnPostFinishedListener* instance. This instance will be called by *AsyncPost* after *CommWrapper* finishes its *post* operation.

A. Job Submission

Mobile applications using *Aneka Mobile Client Library* will commonly follow the lifecycle described by Figure 4 to submit jobs to *Aneka Master* through *Aneka Mobile Client Library*. The main library element is the *TaskWS* class which encapsulates all the communication with *Aneka Task Web Service*. Thus, a mobile application, represented in the figure by the class *AndroidApp*, starts the job submission process by using the *authenticateUser* methods. This method returns the user credentials, which will be required by all the other *TaskWS* methods. After authenticating the user, a mobile application creates the application entity in the Aneka cloud via *createApplication* method, which returns the application id. The FTP server is represented in Figure 4 by the *StorageService* class. The file transfer is done through the *upload* method.

Following the *createAppplication* method, and input files upload if needed, the mobile application is able to submit jobs by using the method *submitJobs*, which returns the job id for each successfully submitted job. This identifier is used to query jobs status via *queryJob* method during their execution. Once the job execution has finished, the *AndroidApp* can download the output files, e.g. a text file for a voice recognition application or a new image for an image filtering one.

B. Asynchronicity

Each of TaskWS's method is accessed in an asynchronous way. The library was purposefully designed this way to facilitate the developers work with Android, as it does not allow network I/O operations in the main thread [5]. This restriction is imposed because network I/O operations in the main thread entails blocking the user interface. In this context, we developed the *AsyncPost* class extending the *AsyncTask* class provided by Android. The latter instantiate a new thread, releasing the application to continue its flow.

Figure 5 shows the sequence diagram for a method called in an asynchronous way. In order to call a method from *TaskWS* class, e.g. *authenticateUser*, *AndroisApp* needs to input a content, e.g. user name and password, and a listener. This input is forwarded to the *AsyncPost* class, that creates a new thread and calls the *post* method from *CommWrapper*. This method encapsulates the network I/O operations and returns the method result to *AsyncPost*. In the case of user authenticate example, it returns the user credential. Finally, the *AsyncPost* calls the listener delivering the result to *AndroidApp*.

Figure 6 details how the exchange of messages between the *Aneka Mobile Client Library* and the *Aneka Task Web Service* is performed. The *CommWrapper* is responsible for encapsulating the process of serializing and deserializing the messages via the *Mapper* class provided by Jackson library as well as communicating to the remote service through the *NetComm* class. Thereby, when the *CommWrapper*'s *post* method is called, it receives a Java object as parameter. This object is serialized and sent to *NetComm*.

The *NetComm* class is responsible for assembling the Uniform Resource Locator (URL), generating the POST request message, and sending it through the *HttpClient* class. When the network operation finishes, the result is returned to *CommWrapper*, which uses again the *Mapper* class in order to deserialize the JSON content to Java object and returns it to *AsyncPost* class.

V. Use Cases and Performance Evaluation

This section shows the Aneka Mobile Client Library's effectiveness through the development of two resource-intensive Android applications, one for image rendering called DroidPov and one for generating the Mandelbrot set called MandelDroid. Furthermore, this section evaluates the MandelDroid application in order to quantify the gain by using the cloud resources in terms of execution time and battery consumption.

[5] http://developer.android.com/reference/android/os/NetworkOnMainThreadException.html

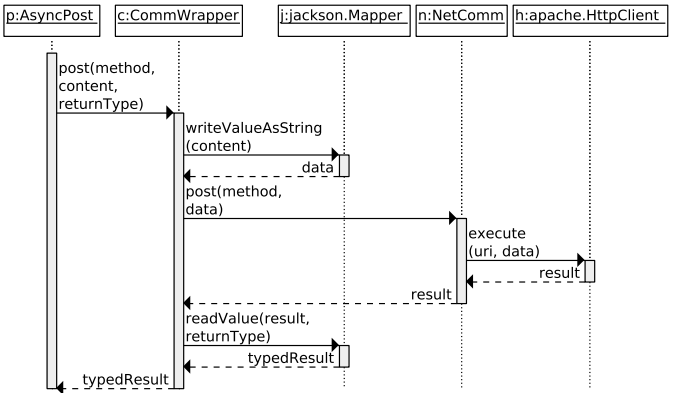
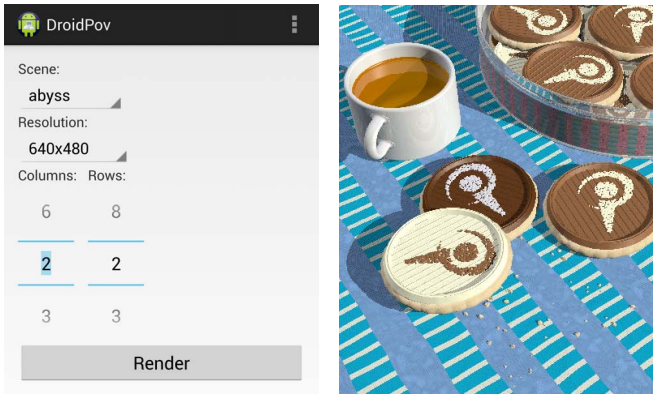
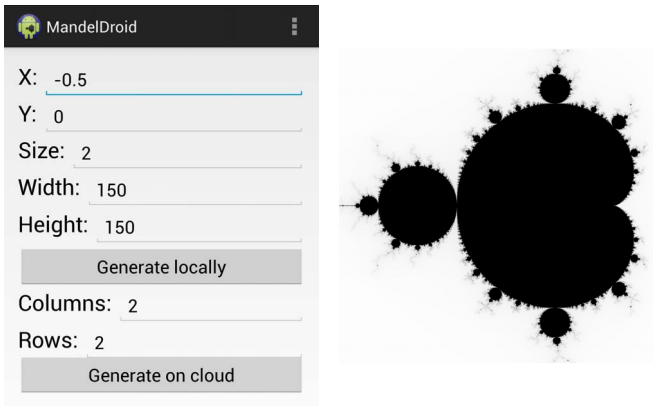


Fig. 6. Sequence diagram for *CommunicationWrapper* class *post* method.



(a) DroidPov *Main Activity*. (b) Scene generated image.

Fig. 7. DroidPov Screens showing (a) the user input parameters: the *Resolution* and the *Scene* to be generated and the number of *Columns* and *Rows* to split the image generation; and (b) the generated image downloaded to the mobile phone.



(a) MandelDroid *Main Activity*. (b) Mandelbrot set generated image.

Fig. 8. MandelDroid screens showing (a) the user input parameters: *X* and *Y* representing the central point, the *Size* of the set, and *Width* and *Height* of the image to be generated; and (b) Mandelbrot set generated image shown in the mobile phone.

A. Use Cases

Alrokayan and Buyya [12] show how the use cases presented in this section benefit from the programming models supported by the Aneka Cloud Application Platform. However, in this section, the same applications are ported to run on Android mobile phones.

DroidPov uses the Persistence of Vision Ray-Tracer (POV-Ray)[6] tool for generating images through the ray-tracing technique [13] from a text file which describes a scene, defining aspects such as light, objects, camera position and atmosphere effect. This file is stored on the mobile phone and sent to Aneka cloud through a FTP server. This type of application has a resource-intensive nature [14]. The delegation approach enables the mobile platform to use the POV-Ray software, available only for Windows platform. Thereby, this approach adds the advantage of allowing the utilization of resources unavailable to the mobile device platform, but that are available in the platform running in the cloud.

As shown in Figure 7a, in order to use the DroidPov, the application user has to set the scene to be rendered, the generated image resolution, and the number of columns and rows to generate the image in parallel. The product of number of rows and columns defines the number of jobs that will be processed by Aneka Workers. Once the image processing has been completed, the image is downloaded and rendered on the mobile phone screen as shown in Figure 7b.

The other developed application is MandelDroid, which consists of a mobile application to generate the Mandelbrot set. This application receives as input a range in the plane of complex numbers. This range is defined by the origin point coordinates and its size, as illustrated in the Figure 8a. The user also specifies the generated image resolution and, if the application runs in the cloud, the number of columns and rows that define how many jobs demanded to split the image generation.

The Mandelbrot algorithm assesses how quickly each point belonging to the range converges to infinity. A gray-scale color is assigned to each point, as shown in Figure 8b. The black points do not converge to infinity, therefore, they belong to the Mandelbrot set. On the other hand, the white points converge to infinity so they do not belong to the Mandelbrot set.

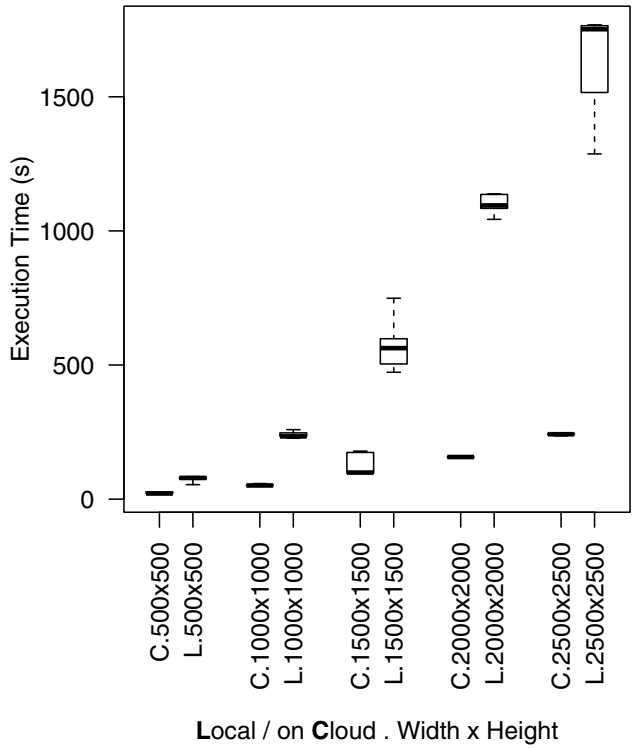
In both applications, the user must configure the connection parameters of Web Aneka Task Service and FTP server, and define a username and password for Aneka authentication.

B. Performance Evaluation

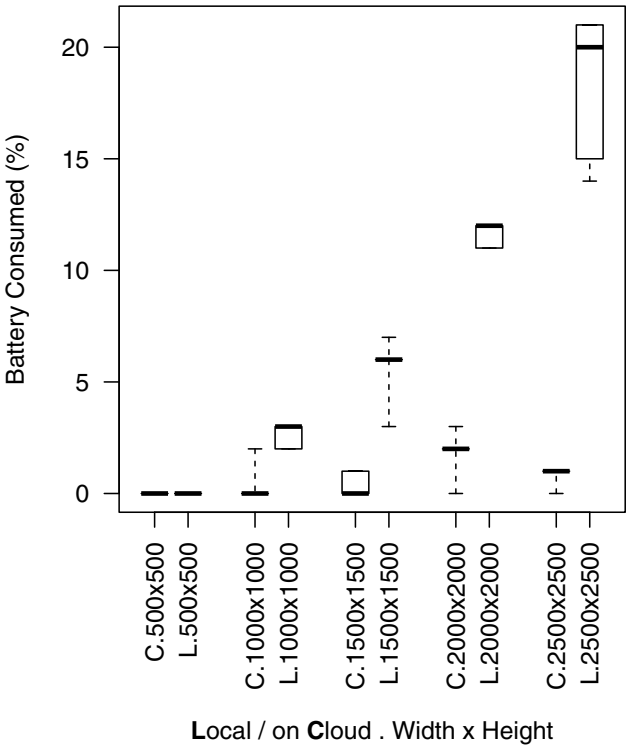
In this section, a performance evaluation is conducted to compare the application execution in a mobile device and in the Aneka cloud and demonstrate the advantage of using the proposed approach, considering the execution time and battery consumption metrics. Since the POV-Ray software is not available for Android platform, only the MandelDroid application is employed in the experiments.

The first experiment consists of executing the MandelDroid application, both on the mobile phone and on the cloud, for generating Mandelbrot set images with 5 different resolutions: 500x500, 1000x1000, 1500x1500, 2000x2000 and 2500x2500. The origin point was fixed in (-0.5, 0) and the range size to 2.

[6]http://www.povray.org/



(a) Execution time, in seconds, for each parameter combination.



(b) Percentage of battery consumed for each parameter combination.

Fig. 9. Boxplot graphs comparing the results for *L*ocal and on *C*loud execution with different image resolutions: 500x500, 1000x1000, 1500x1500, 2000x2000 and 2500x2500. Each combination of parameters was executed 5 times.

Also, the application was split in 4 jobs in order to execute on mobile device and Aneka cloud. Each experiment was executed 5 times, resulting in a total 60 rounds. The observed metrics were battery consumption, in percentage, and execution time, in seconds, both monitored using the Android API.

The experimental scenario is composed of a Asus Padfone Infinity a86 (T004) mobile phone with Android 4.2.2, 32 GB of storage, 2 GB RAM, support WLAN 802.11a/b/g/n/ac and CPU Snapdragon 800 quad-core (2.2GHz) and one Azure Standard tier A3 virtual machine, with 4 2.1GHz cores and 7 GB memory, running Windows Server 2012 R2. The internet connection used to connect the mobile device to FTP server and the Aneka Task Web Service has upload and download rates of 12.11 Mbps and 11.79 Mbps, respectively.

Figure 9a presents the MandelDroid local (L) and on cloud (C) execution time with the different image resolutions. This plot shows that the time spent for the application execution on cloud is lesser for all the resolutions. This difference is bigger as the resolution increases, representing an economy of up to 87% for the 2500x2500 resolution.

Figure 9b presents the battery consumption during the experiment execution, where it can be observed that the consumption showed by the mobile device is bigger for all resolutions except 500x500. This result is expected, whereas during the execution on cloud, the mobile device uses energy only to wait for the remote execution result, for instance, to keep the Wi-Fi connection and display active. For the 500x500 resolution, the required processing to generate the image is low, both using the cloud or not, which represents a negligible impact to battery consumption. The battery savings can reach 95.23% for the 2500x2500 resolution.

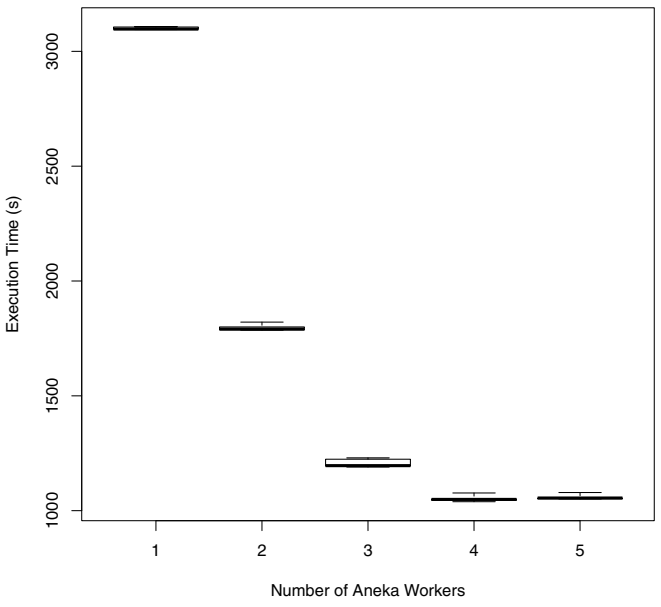


Fig. 10. Boxplot graph comparing the impact of the number of Aneka Workers on execution time. The image resolution is fixed to 12500x12500 and the image generation was split in 25 jobs.

The second experiment aims to evaluate the impact of different numbers of Aneka Workers on the execution time. This

experiment scenario differs from the first because c3.xlarge Amazon EC2 virtual machines were used instead of Azure ones. Each virtual machine has 4 2.8GHz cores and 7.5 GB and runs Windows Server 2008 R2.

Figure 10 presents the execution time for generating a Mandelbrot set image with 12500x12500 resolution. The image generation is split in 25 jobs and distributed among workers that vary from 1 to 5 and each boxplot in the figure corresponds to 5 application execution rounds. For each worker added to the experiment, the total number of CPU cores increases by 4, varying from 4 to 20 cores.

By increasing the number of workers from 1 to 3, the execution time is significantly reduced. However, an expressive gain is not observed when the number of workers increases from 4 to 5. Two different situations can explain this: (i) each of the 25 jobs have similar execution time and at least one core needs to execute 2 jobs; or (ii) the execution time of the jobs are different and the longest jobs are limiting the application execution time. In this context, the ideal number of workers is 4, considering the image resolution and the number of jobs aforementioned and having the reduction of execution time as main goal.

VI. Conclusions and Future Work

Currently, developers are facing complex mobile applications that require accessing distributed clouds. The development of these applications is challenging because it involves dealing with different cloud providers Web APIs and mobile platforms. Moreover, porting these APIs to mobile devices is a difficult task due to compiler limitations, additional dependencies, and code incompatibility. In order to reduce the effort and complexity of developing mobile cloud applications, the Aneka Mobile Client Library was proposed and described in this paper. This library encapsulates the processes of connecting to cloud, serializing and deserializing messages, sending messages, and collecting their responses.

A mobile cloud architecture was also proposed to delegate resource-intensive mobile tasks in an asynchronous manner in order to alleviate the mobile device load and, consequently, extend the battery life. This architecture was designed to leverage the Aneka PaaS solution, which provides transparent resource provisioning and job scheduling services and encapsulates different cloud providers Web APIs. Thereby, the user has no concern in allocating or deallocating virtual machines or distributing the jobs among the resources.

This paper also investigated the effectiveness of the Aneka Mobile Client Library through the development of two resource-intensive mobile applications: ray tracing image generation and Mandelbrot set generation. A performance evaluation was conducted and the results showed the feasibility of the architecture, since the Aneka cloud spends less time to execute the MandelDroid application, representing a reduction of up to 87% of execution time while reducing battery consumption by up to 95.23%.

As future work, we are planning to (i) improve the verification process of job execution by integrating with Android push notification service, avoiding the battery consumption intrinsic to the pooling approach currently used to check the status of submitted jobs; (ii) port the Aneka Mobile Client Library for other mobile platforms such as iOS and Windows Phone; and (iii) design new scheduling and provisioning policies that consider user preferences, such as budget and application execution deadline, and mobile context parameters, such as battery level and internet connection type (WiFi of mobile).

Acknowledgements

This work is supported by the Australian Research Council through Future Fellowship program. We would like to thank Amir Vahid Dastjerdi, Nikolay Grozev, Rodrigo N. Calheiros, Satish Narayana Srirama, and Deborah Magalhães for their comments on improving the quality of the paper.

References

[1] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, vol. 43, no. 4, pp. 51–56, 2010.

[2] S. Abolfazli, Z. Sanaei, M. Alizadeh, A. Gani, and F. Xia, "An experimental analysis on cloud-based mobile augmentation in mobile cloud computing," *Consumer Electronics, IEEE Transactions on*, vol. 60, no. 1, pp. 146–154, 2014.

[3] M. Shiraz, A. Gani, R. H. Khokhar, and R. Buyya, "A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 3, pp. 1294–1313, 2013.

[4] M. Satyanarayanan, "Fundamental challenges in mobile computing," in *Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing*. ACM, 1996, pp. 1–7.

[5] S. Abolfazli, Z. Sanaei, and A. Gani, "Mobile cloud computing: A review on smartphone augmentation approaches," *arXiv preprint arXiv:1205.0451*, 2012.

[6] H. Flores and S. N. Srirama, "Mobile cloud middleware," *Journal of Systems and Software*, 2013.

[7] S.-H. Hung, T.-W. Kuo, C.-S. Shih, J.-P. Shieh, C.-P. Lee, C.-W. Chang, and J.-W. Wei, "A cloud-based virtualized execution environment for mobile applications," *ZTE Communications*, vol. 9, no. 1, pp. 15–21, 2011.

[8] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: making smartphones last longer with code offload," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*. ACM, 2010, pp. 49–62.

[9] C. Vecchiola, X. Chu, and R. Buyya, "Aneka: a software platform for .NET-based cloud computing," *High Speed and Large Scale Scientific Computing*, pp. 267–295, 2009.

[10] R. N. Calheiros, C. Vecchiola, D. Karunamoorthy, and R. Buyya, "The Aneka platform and QoS-driven resource provisioning for elastic applications on hybrid clouds," *Future Generation Computer Systems*, vol. 28, no. 6, pp. 861–870, June 2012.

[11] K. Hameseder, S. Fowler, and A. Peterson, "Performance analysis of ubiquitous web systems for smartphones," in *Performance Evaluation of Computer & Telecommunication Systems (SPECTS), 2011 International Symposium on*. IEEE, 2011, pp. 84–89.

[12] M. Alrokayan and R. Buyya, "A web portal for management of Aneka-based multicloud environments," in *Proceedings of the Eleventh Australasian Symposium on Parallel and Distributed Computing-Volume 140*. Australian Computer Society, Inc., 2013, pp. 49–56.

[13] A. S. Glassner, *An introduction to ray tracing*. Morgan Kaufmann, 1989.

[14] C. Mateos, A. Zunino, M. Hirsch, M. Fernández, and M. Campo, "A software tool for semi-automatic gridification of resource-intensive java bytecodes and its application to ray tracing and sequence alignment," *Advances in Engineering Software*, vol. 42, no. 4, pp. 172–186, 2011.

Privacy Preserving Third Party Auditing In Multi Cloud Storage Environment

M.S.Shashidhara
Prof. & Head ,Computer Applications
The Oxford College of Engineering
Bangalore,Karnataka ,India

Jaini.C.P
Research Scholar
Bharathiar University
Coimbatore,Tamilnadu,India

Abstract. The on-demand, pay-per-use, and scalable services provided in cloud model guarantee to reduce capital as well as running expenditures for both hardware and software. In cloud environment, users can remotely store their data and access them from a shared pool of configurable computing resources, without local data storage burden. We discuss various methods related to the security and privacy capabilities in cloud paradigm especially data storage in multi cloud environment. We provide three models in form of multicloud architectures which allow categorizing the schemes and analyze them according to their security benefits. The different methods include, resource replication, split application system into tiers based on PIR methods, split both application logic and data into segments. In addition, since the integrity protection of data is a fearsome task in Cloud computing for users with limited computing resources, vulnerabilities in user data privacy are also possible in third party auditing. So we propose a safe cloud storage methodology which supports privacy-preserving third party auditing. And we study the outcomes to perform audits concurrently for multiple users in an efficient manner. Experimental results show that the third party auditing computation time is better than existing approach.

Keywords: Mutual authentication, Identity management, Access control, Distributed computing, Security analysis.

I. INTRODUCTION

Clouds can be categorized into public, private and hybrid. Third-party service providers offer public cloud which involves resources outside the users' location. If the cloud system is set up on the users' location (data centers) it is termed as private cloud. Combined approach of both types is denoted as hybrid cloud. This paper concentrates on public clouds where services demand for the highest security requirements. Also it includes high potential for security prospects.

Public clouds, includes the IaaS, Paas and SaaS cloud service layers. The users data are transferred from inside to outside the organization in this cloud environment. In this case, a number of issues among which security aspects are created when considering cloud computing [1].

So to reduce the risk for data and applications in a public cloud, multiple clouds can be used. Several approaches are proposed in this regard which differ in sharing and delivery methods, hardware and software resource allocation, encryption and decryption methods as well as different security levels.

Cloud computing paradigm also faces the problem during secure outsourcing of sensitive data and processes. When taking into account using a cloud service, control and protection left out for the data given to the cloud provider.

So this paper suggests that the security can be increased if the architecture is changed from single cloud to multi cloud environment. This paper provides three models which can be applied in multi-cloud architectures. These multi-cloud architectures classifies the given schemes and analyzes them based to their security levels.

In addition, security mechanisms involved during third party auditing of outsourced data is discussed. How the enhancements required in integrity checking is also studied. Like the previous work carried out in [2], our work also focuses on how to carry out the privacy-preserving public auditing related with data storage in cloud computing. Since the individual auditing is tedious, we need to consider performing multiple auditing tasks simultaneously. We also study the methods to perform the auditing without demanding the local copy of data and thus drastically reduce the communication and computation overhead.

The rest of this paper is organized as follows: Section 2 motivates the need for cloud security measures by briefly reviewing the present methods. The observations show that most of the methods do not consider the precise properties of the clouds themselves. To categorize and evaluate these proposals, three distinct multi-cloud architectures are proposed. The security in multi cloud architecture is discussed in Section 3. The proposed schemes are introduced in Section 4 including architecture and flowcharts and general discussions are presented in Section 5. Section 6 provides conclusion.

II. LITERATURE REVIEW

Different people understand Cloud in different ways. The most inexpensive way to acquire IT services is the public cloud. The main characteristic of public cloud is that it is reasonably cost effective "pay-to-use" model.

Jens-Matthais Bohli et al. [3] stated that security challenges are still among the major obstruction when

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

considering the adoption of cloud services. This generates a lot of research activities, resulting in more proposals targeting a range of cloud security threats. Alongside with these security issues, the cloud models come with a new set of unique features, security approaches and architectures [3].

One of the major problems that the cloud computing contains is that of secure outsourcing of business-critical information and applications. The user must know that the entire data given to the cloud provider is out of the user's control and protection when considering using a cloud service. The cloud provider gains total control of these information and applications, if deploying to the cloud through IaaS or PaaS. So, the main prerequisite in cloud computing is a well-built trust relationship between the cloud provider and the cloud user.

Usually, the attacker accesses the cloud data by taking snapshots of storage. He/She modify the data in the storage one time, many times, or continuously. Also, the processing logic of the cloud, the functionality, input/ output data can be changed by him/her. Even though it is being assumed a cloud provider is honest, the malicious employees can make successful attacks.

Google Docs [4] that allows users to process text or spreadsheet documents and presentation online and share them with others. An incident in a SaaS cloud occurred in 2009 with Google Docs. This system had a defect such that once a document was shared with anyone, every user (previously the document owner has ever shared documents with before) can access the file.

No additional knowledge is required to get unauthorized access to secured information because of this technical malfunction. Major cloud providers' cloud systems are subjected to many attacks since they contain rigorous security defects in different kinds of clouds (see [5]).

If an attacker penetrates the cloud system, the entire users' all information and applications on that cloud system may become subject to malicious actions immediately. So the cloud computing methods requires a detailed analysis of how to prevent them. The invasion immediately affects the accessibility, integrity, and confidentiality. The terms and conditions regarding information and applications can be breached directly or indirectly. Many research activities targeting the various cloud security threats are also being conducted [6-8].

M.K.M. Reiter et al. [9] discussed some attack techniques for virtualization of the Amazon EC2 IaaS service. In their approach, new virtual machines are allocated by the attacker until one runs on same hardware as the victim machine. Then, he/she performs cross-VM side-channel attacks to know and change the victim's information.

The encryption methods alone cannot prevent data flowing towards external parties without a well designed auditing methodology. The key management itself will not solve the data privacy problem. Due to the exposure of decyption keys, data leakage will still remain. Introducing a privacy-preserving third-party auditing, without data encryption, is the problem to be considered. Our work is focusing in this area especially in multi-cloud environment. Using our methodology, we guarantee that the auditors could not learn any information about the content stored in the cloud nodes during the auditing process.

Like [3], for Privacy-preserving: we ensure that the auditors cannot extract users' content from the collected information during the auditing process. It works in light-weighted manner by allowing auditors to audit with minimum computation overhead.

The auditing process can concurrently access the file segments located in multiple cloud servers in secure and efficient manner by large number of different users simultaneously.

The auditing task can be expensive for cloud users if the outsource data's size is larger [10]. The operating cost of cloud storage should be minimum such that user perform less operations to use/retrieve the data.

III. SECURITY IN MUTI CLOUD ARCHITECTURE

Bernstein et al. [11] proposed the idea of multiple clouds usage. However, they did not concentrate security. Some other approaches consider the security effects which are operating on different cloud service levels. They are partly combined with cryptographic techniques and applied in different usage scenarios. By combining different clouds, the security can be improved. It makes the attacker much harder to extract the whole data and applications.

We provide three methods related to the security capabilities especially data storage in multi cloud environment. Three models are provided and their security benefits are analyzed. Resource replication, PIR based segmentation of application system into tiers and segmentation of both application logic and data are provided which stores/ processes data and application in multiple cloud nodes of different cloud providers.

The proposed methods are applicable to different application scenarios and provide respective security merits. In addition, if the methods are combined, then they result in combined security benefits. The easy deployment of above methods is explained in following sections and details for more compromised cloud systems are also presented.

IV. PROPOSED SCHEMES

A. RESOURCE REPLICATION

The concept and application designed details for resource replication are explained here. Even though the terms and conditions are respected by all participants, the cloud environment possesses the risk of compromising by third parties. To avoid those problems, if multiple copies of same data or application is being replicated and executed in multiple clouds, then the usage statistics of those data cannot be tracked by any single or combined cloud providers. The client application if uploads the same application under different names in multiple clouds and access any one of them at different time intervals, then the cloud provider lacks the knowledge of access statistics of a given particular data or application. Moreover, after combining the results of same applications executed in various cloud servers, the user can ensure the data integrity.

One more advantage of resource replication is that, the robustness of the system is increased against system failures [12]. Moreover, this will increase the

communication flexibility in terms of business processes. Storage overhead is accepted if the results are better than single cloud environment. If the client application changes the file name and size of the same application and uploads them in multiple cloud nodes, then the cloud providers would not be able to monitor and verify the similarity in intermediate results of data values produced by those applications.

The practical implementation of the above mentioned application is developed as Windows application using Visual Studio .Net 2005 with C# 2.0. The architecture is shown in Fig. 1 and Flowchart is shown is Fig. 2.

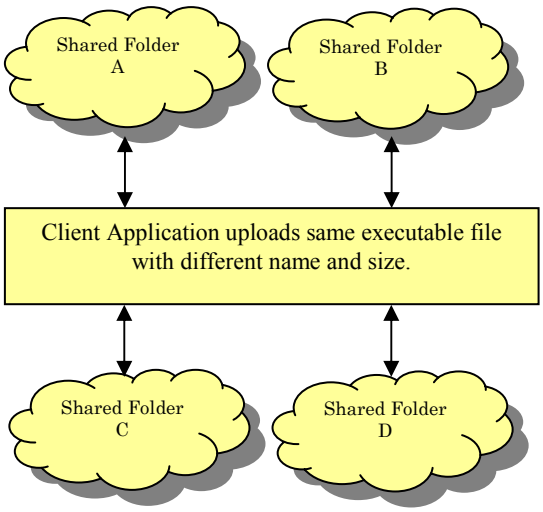


Fig. 1. Resource Replication.

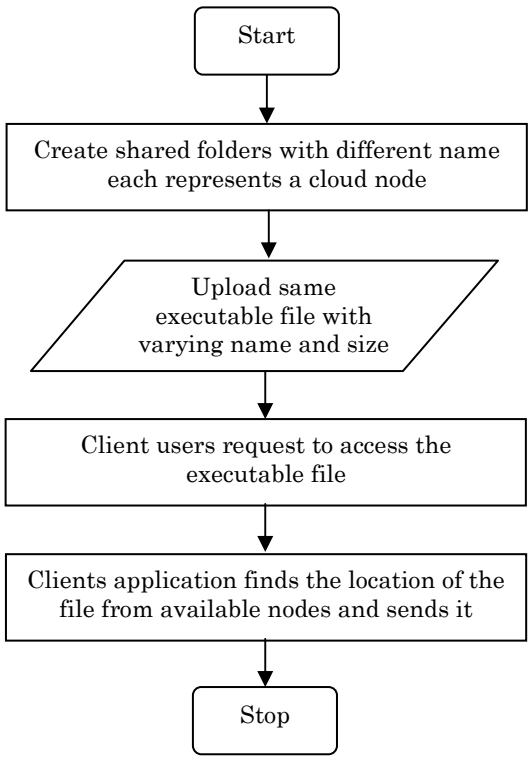


Fig. 2. Flowchart for Resource Replication.

B. PIR BASED SEGMENTATION OF APPLICATION SYSTEM INTO TIERS

PIR based segmentation of application system into tiers is discussed here. We present an application which contains client tier, web service tier and database tier to simulate the method. Since Private Information Retrieval makes the users to access the data from database without revealing the record information, this method is adopted here. Generally, the web session object can be hacked by attackers and the content about the user is known to them. For safe access of the data in the database, in this method, we design the client application such that a prime number is associated for each record number in the database.

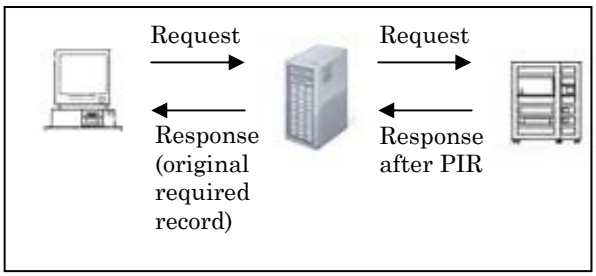


Fig. 3. PIR based Segmentation of application system into tiers.

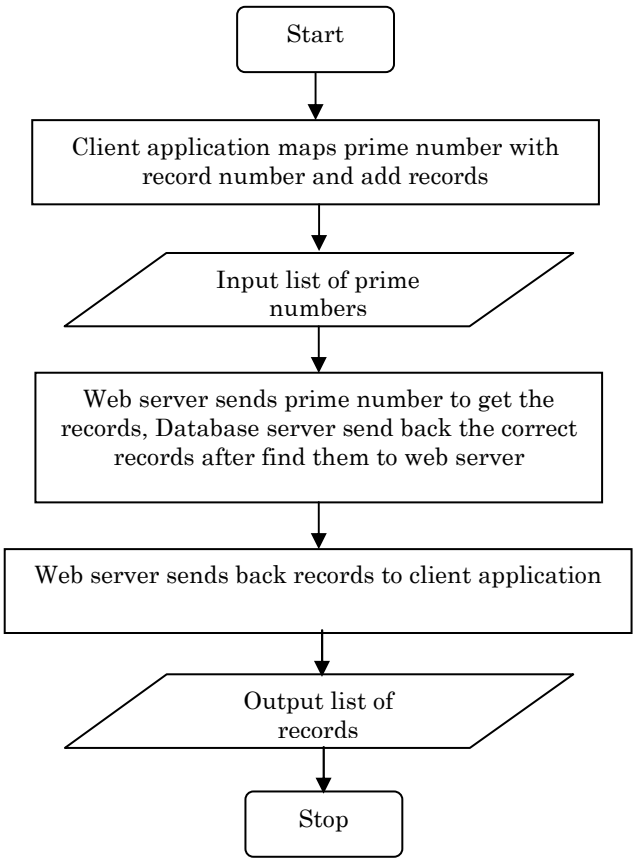


Fig. 4. Flowchart for PIR based Segmentation of application system into tiers.

During the data access, client sends a list of prime numbers to the web server. The web server sends the query with prime numbers only. The stored procedure in database find outs all the related record number and data values of those records and send back to web server. From the web server, the records are given to client application.

This eliminates the session object to contain the record number and the related data and thereby security is increased. The process is shown in Fig. 3 and the flowchart is shown in Fig. 4.

C. SEGMENTATION OF APPLICATION LOGIC AND DATA

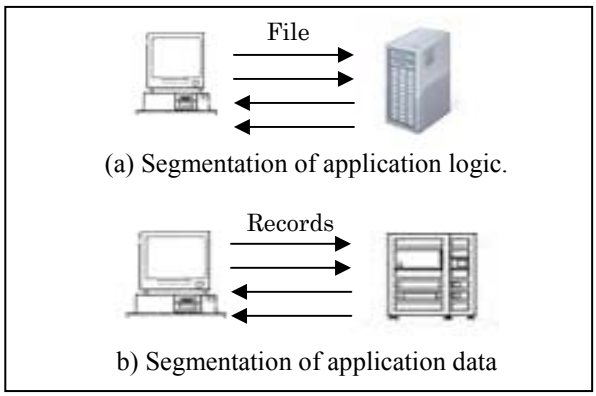


Fig. 5. a) Segmentation of application logic. (Two arrows indicated to show that two modified instances of same executable application are uploaded/ downloaded). b) Segmentation of application data. ((Two arrows indicated to show that two types of records are saved/retrieved in two different databases).

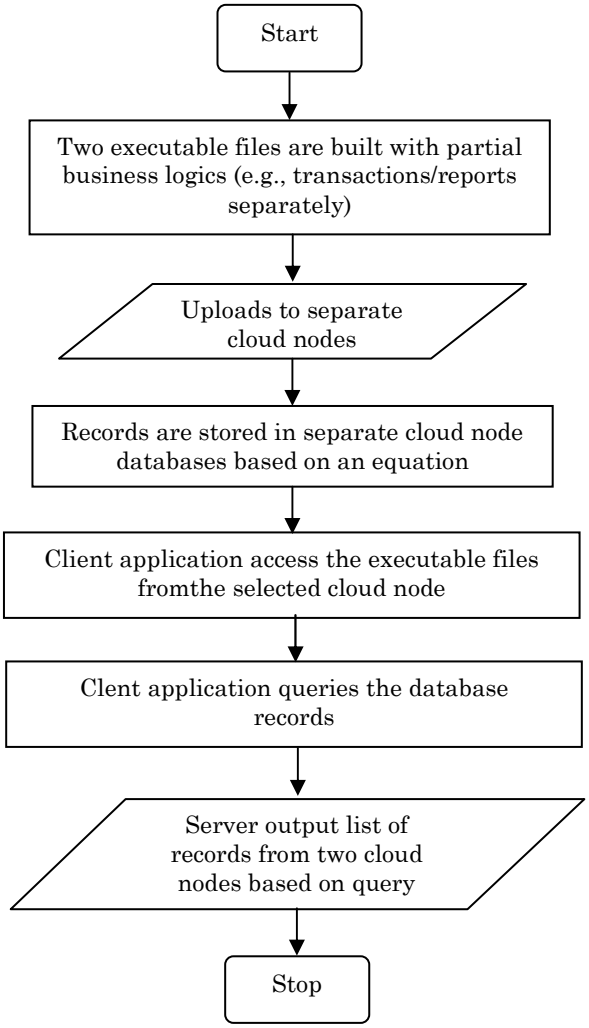


Fig. 6. Flowchart - Application logic/data Segmentation.

Segmentation of application logic and data is discussed here. We present an application in which the different versions of the file are stored in different client nodes. For example, same executable file (a sample accounting application) with options containing transaction entries (A) is built and the version is uploaded to one cloud node. The same executable file with options containing report generation (B) is built and that version is uploaded to other cloud node in different name than the previous one.

Depending on the requirement, the requirement executable file (A or B) is requested and accessed. This makes the application logic to split into two segments and accessed from separate cloud nodes. So the whole application logic is not known to a single cloud provider. Moreover, the data is also saved into two (or more) databases in which records of same application are split and saved in them. For example, based on an equation, the records are split. Example scenario: 1) Records with odd serial numbers (or unique identifiers) in one cloud node's database (D1) and even serial numbers in other cloud node's database (D2). 2) Records with serial numbers (1,2,3,4,11,12,13,14,…) in D1 and (5,6,7,8,9,10,15,16,17,18,19,20…) in D2. 3) Records with multiples of 3 in D1 and other records in D2.

Likewise, records can be split using any logic implemented in client application. The client application takes care of fetch the records from selected cloud nodes and consolidates the collected records. So a single cloud provider cannot access all the data which protects the data confidentiality.

The practical implementation uses two cloud nodes with separate shared folders and separate databases to simulate the segmentation of logic as well as data. The architecture is shown in Fig. 5 and Flowchart is shown is Fig. 6.

D. SECURED THIRD PARTY AUDITING SCHEME

Secured third party auditing scheme is discussed here. How to support privacy-preserving third party auditing without accessing all the part of the data is explained.

We present an application which is integrated with the previous method (segmentation of application logic). The steps to operate the application are: 1) A document (e.g., executable file, text or binary file) (A or B in previous method) is chosen and the file is split into 'n' segments (say 10 segments). 2) Unique prime numbers are generated and assigned for each segment. 3) Starting and ending bytes of each segment are extracted and encoded. The file id, prime numbers, starting and ending bytes are given to third party auditing process. During auditing, the party enters some segment ids, their prime numbers, starting and ending encoded bytes so that the process verifies the file content stored in the cloud nodes is not modified or corrupted. The integrity is found to be verified properly if the credentials matched and the file content matches with decoded bytes in corresponding locations.

For a given executable file (A and B in two separate nodes), two sets of credentials need to be given to third party auditing process, so that two cloud nodes need to be checked separately and file/ data integrity is verified.

The practical implementation uses two cloud nodes with separate shared folders and files stored in them. A separate third party auditing application is presented to simulate the auditing process to verify the file integrity. The architecture is shown in Fig. 7 and Flowchart is shown is Fig. 8.

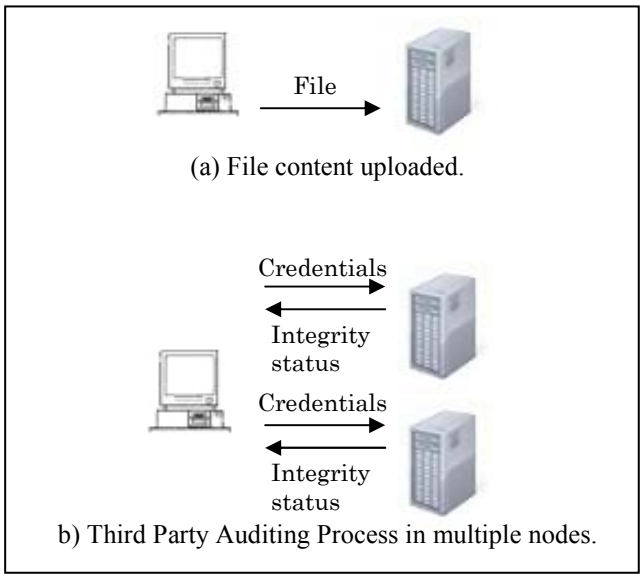


Fig. 7. a) File content uploaded. b) Third Party Auditing Process in multiple nodes.

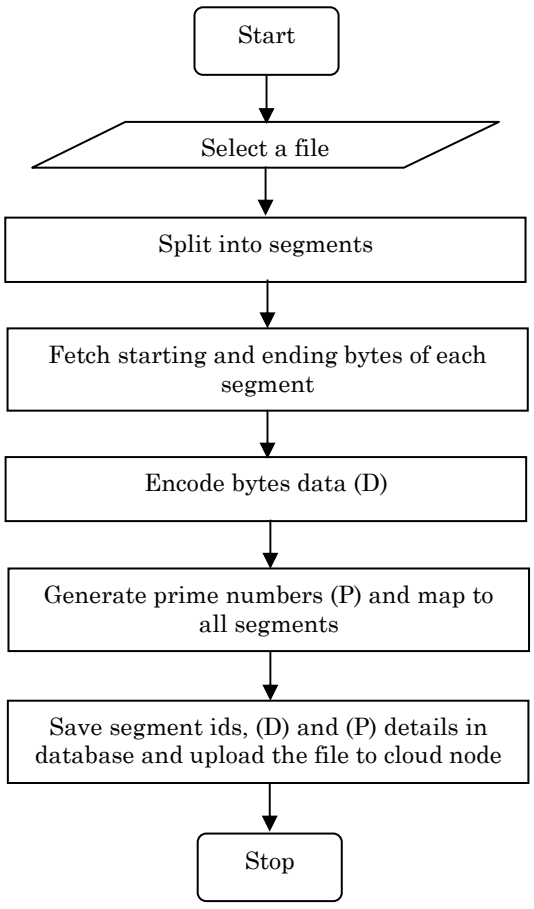


Fig. 8. a) Upload File content to cloud node's folder and save file credentials in cloud database.

The security in multi cloud architecture is discussed above.

V. DISCUSSION

The storage schemes that can be adopted are explained above along with secure third party auditing. We present three methods of data storage and one auditing process which improve security in multi cloud architectures. The replication is required in first scheme only and nodes where computation is required and its complexity is listed in the following table.

TABLE. 1. REPLICATION REQUIREMENT AND COMPUTATION OVERHEAD

METHOD	REPLICATION	COMPUTATION OVERHEAD
Resource Replication	Required	In client only
PIR based segmentation	Not required	Low in client tier/ More in database tier (stored procedure) and negligible in web tier
Segmentation of application logic and data	Not required	In client only
Third party auditing	Not required	High In client, Low in third party system and negligible in cloud node

The above table (Table. 1) shows the replication requirement in cloud nodes and computational overhead in client/cloud node/cloud database and third party system. The figure (Fig. 9) shows the file content and credentials. The figure (Fig. 10) shows the file id generated. The file is uploaded to cloud node and credentials are given to third party for auditing process. During the auditing process, the third party should enter cloud node id, file id and credentials (random segment ids, their prime numbers, encoded starting and ending bytes.

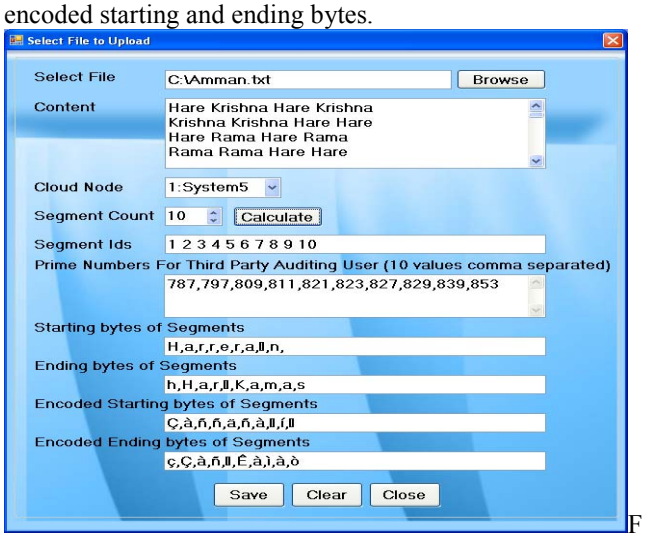


Fig. 9. File content and credentials

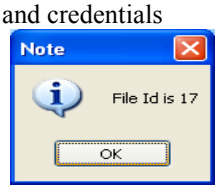


Fig. 10. File Id generated after file upload.

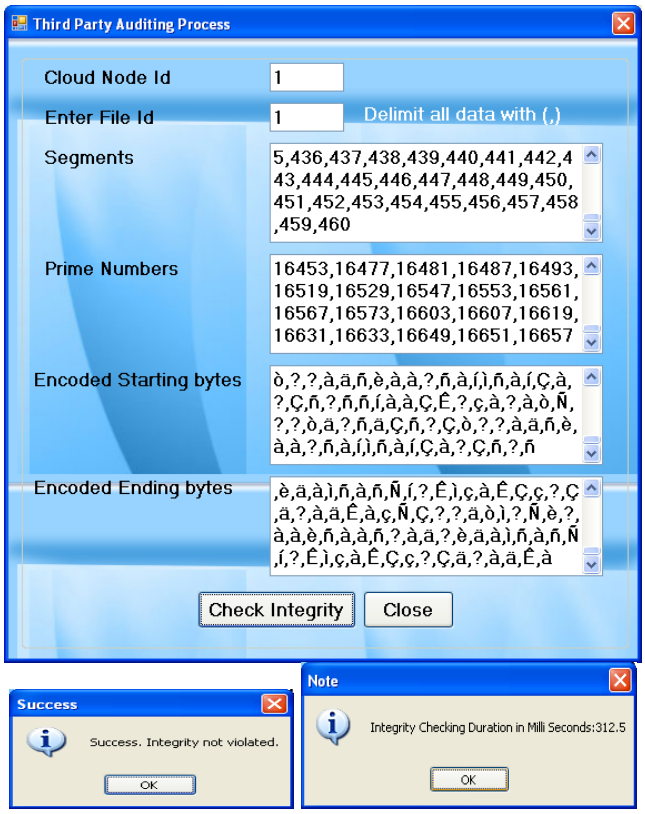


Fig. 11. Third party auditing process.

The above figure (Fig 11.) shows the credentials to be given during the third party auditing process.

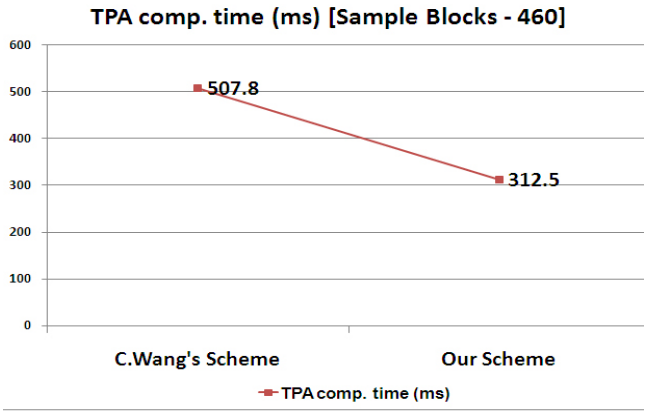


Fig. 12. Chart comparison for TPA computation time (C.Wang;s scheme vs Our scheme) (Chart Type – Line Chart).

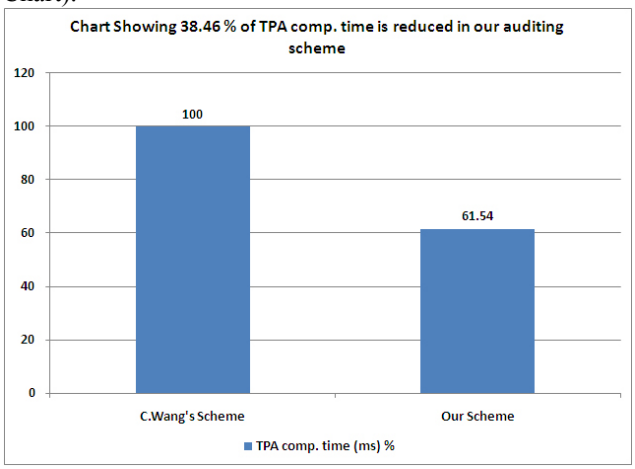


Fig. 13. Chart comparison for TPA computation time in %.

VI. CONCLUSION

We present three schemes that can be applied in multi cloud environment to increase the security aspects. Hiding resource usage statistics of a single resource for a single cloud provider is achieved if first method is applied. The computation and data transfer size is very low if the second method is applied. The third method provides the security such that a single provider may not be aware of the execution flow of the single application as well as the cloud provider could not know or access all the data. The fourth method provides the benefit of auditing with very low credential data to verify the file content. We proved that the third party auditing computation time is better than existing approach. In future, our study will be focusing on security proof and enhancements in data retrieval of the proposed framework.

REFERENCES

[1] F. Gens, "IT Cloud Services User Survey, pt.2: Top Benefits & Challenges," blog, http://blogs.idc.com/ie/?p=210, 2008.

[2] Cong Wang, Sherman S.-M. Chow, Qian Wang, Kui Ren and Wenjing Lou, "Privacy-Preserving Public Auditing for Secure Cloud Storage" IEEE TRANSACTIONS ON CLOUD COMPUTING YEAR 2013.

[3] Jens-Matthias Bohli, Nils Gruschka, Meiko Jensen, Luigi Lo Iacono, and Ninja Marnau, "Security and Privacy-Enhancing Multicloud Architectures" IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL. 10, NO. 4, JULY/AUGUST 2013.

[4] J. Kincaid, "Google Privacy Blunder Shares Your Docs without Permission," TechCrunch, http://techcrunch.com/2009/03/07/huge-google-privacy-blunder-shares-your-docs-without-permission/, 2009.

[5] J. Somorovsky, M. Heiderich, M. Jensen, J. Schwenk, N. Gruschka, and L. Lo Iacono, "All Your Clouds Are Belong to Us: Security Analysis of Cloud Management Interfaces," Proc. Third ACM Workshop Cloud Computing Security Workshop (CCSW '11), pp. 3-14, 2011.

[6] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "Hey, You, Get Off of My Cloud: Exploring Information Leakage in Third-Party Compute Clouds," Proc. 16th ACM Conf. Computer and Comm. Security (CCS '09), pp. 199-212, 2009.

[7] N. Gruschka and L. Lo Iacono, "Vulnerable Cloud: SOAP Message Security Validation Revisited," Proc. IEEE Int'l Conf. Web Services (ICWS '09), 2009.

[8] M. McIntosh and P. Austel, "XML Signature Element Wrapping Attacks and Countermeasures," Proc. Workshop Secure Web Services, pp. 20-27, 2005.

[9] Y. Zhang, A. Juels, M.K.M. Reiter, and T. Ristenpart, "Cross-VM Side Channels and Their Use to Extract Private Keys," Proc. ACM Conf. Computer and Comm. Security (CCS '12), pp. 305-316, 2012.

[10] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in Proc. of ESORICS'09, volume 5789 of LNCS. Springer-Verlag, Sep. 2009, pp. 355–370.

[11] D. Bernstein, E. Ludvigson, K. Sankar, S. Diamond, and M. Morrow, "Blueprint for the Intercloud—Protocols and Formats for Cloud Computing Interoperability," Proc. Int'l Conf. Internet and Web Applications and Services, pp. 328-336, 2009.

[12] I. Koren and C.M.C. Krishna, Fault-Tolerant Systems. Morgan Kaufmann, 2007.

Secure Data Mining in Cloud using Homomorphic Encryption

Deepti Mittal, Damandeep Kaur, Ashish Aggarwal
Computer Science Engineering Department
Thapar University
Patiala, India 147001
Email: deep.m88@gmail.com, damandeep.kaur@thapar.edu, ashish.aggarwal@thapar.edu

***Abstract*— With the advancement in technology, industry, e-commerce and research a large amount of complex and pervasive digital data is being generated which is increasing at an exponential rate and often termed as big data. Traditional *Data Storage* systems are not able to handle *Big Data* and also analyzing the *Big Data* becomes a challenge and thus it cannot be handled by traditional analytic tools. *Cloud Computing* can resolve the problem of handling, storage and analyzing the *Big Data* as it distributes the big data within the cloudlets. No doubt, *Cloud Computing* is the best answer available to the problem of *Big Data* storage and its analyses but having said that, there is always a potential risk to the security of *Big Data* storage in *Cloud Computing*, which needs to be addressed. Data Privacy is one of the major issues while storing the *Big Data* in a Cloud environment. Data Mining based attacks, a major threat to the data, allows an adversary or an unauthorized user to infer valuable and sensitive information by analyzing the results generated from computation performed on the raw data. This thesis proposes a secure k-means data mining approach assuming the data to be distributed among different hosts preserving the privacy of the data. The approach is able to maintain the correctness and validity of the existing k-means to generate the final results even in the distributed environment.**

Keywords*— *cloud computing, Security, k-means, data mining, encryption.

I. INTRODUCTION

Cloud computing refers to the web-based computing, providing users or devices with shared pool of resources, information or software on demand and pay per-use basis. It frees a user from the concerns about the expertise in the technological infrastructure of the service. It allows end user and small companies to make use of various computational resources like storage, software and processing capabilities provided by other companies. The cloud services can be divided into three categories: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS)[2]. Amazon, Microsoft, Google are some of the major cloud service providers. Google App Engine (GAE) is a type of PaaS provided by Google which allows web application hosting. Windows Azure, SQL Azure is some of the services offered by Microsoft providing processing and storage capabilities for large datasets [3]. Amazon Web Services (AWS) including Simple Storage Service (S3), SQS, EC2 are cloud services provided by the Amazon [1]. Thus convenience, on demand measured access, shared easily configurable computational resources, rapid provisioning, location independence and self-service are some of the major characteristics of a cloud environment [2].

Despite all the above powerful functionalities provided by the cloud computing techniques, a lot of perspective customers and users lack interest for cloud services. Reason being the cloud issues which includes: availability or business continuity, Data confidentiality, data transfer bottlenecks, performance unpredictability, scalable storage, bugs in large distributed systems, scalability, reputation fate sharing and software licensing [3]. Of all the above issues Security comprising of data privacy issue or confidentiality of data is one of the major. As all the data resides with the cloud provider, a serious data privacy issue arises if the provider misuses the data or the information. Also any attacker or adversary having an unauthorized access to the storage on cloud can mine the data and retrieve large amount of confidential data. Various data analysis techniques or algorithm are available today which can be used successfully to mine valuable information from the large datasets by analyzing the behavioral and statistical data. Many cloud providers offer these data mining facilities to users which can be used by an adversary. Google also uses some data mining technique to predict search results by analyzing the user behaviors [4]. So, data mining can be a serious threat to the cloud security. Specially, to the organizations dealing with the financial, governmental, education or legal issues of people, leaking of which can sometime result in national catastrophes for e.g. collection of financial, health etc information by TIA (Total Information Awareness) in 2002 [5] and analysis of phone records of people gathered from phone companies by NSA for identifying the possible terrorists in May 2006[5]. Also, according to a survey conducted by Rexer analytics, 7% of the data miners analyze the data using the cloud [6], due to the cheap and elastic computing powers offered by the cloud computing. So, maintenance of client privacy goes in parallel with data privacy in cloud and is a major area of concern for the cloud provider as well as cloud user.

This paper presents an approach to mine the data securely using k-means algorithm from the cloud even in the presence of adversaries. This approach assumes that the data is not stored in a centralized location but is distributed to various hosts. This proposed approach prevents any intermediate data leakage in the process of computation while maintaining the

978-1-4799-6141-2/14/$31.00 ©2014 IEEE

correctness and validity of the data mining process and the end results.

II. Related Work

Preserving the privacy of the data mining algorithm has been a concern of researchers for long and a number of algorithms have been proposed for the same. [7] Focuses on improving the security of two-party k-means while maintaining the correctness of algorithm. K-anonymity [10], noise transformation and multiplicative transformation [9][17] are some PPDM(privacy preserving data mining) methods. Compared to PPDM secure cloud mining is a relatively newer field. [15] is a detailed survey of the key security challenges faced by the developers while designing the cloud application, privacy risks to the Cloud, the various Privacy Requirements and finally gives the design guidelines for developers to tackle the issue of privacy. [4] proposes an extended solution to the current techniques using trusted computing and various new, modified cryptographic techniques for privacy enhanced business intelligence. The attacks in a Cloud Data Mining system can be listed as *DoS* (Denial of Service) attack, *DDoS* (Distributed Denial of Service), *Sniffing, DNS attack, Man in the Middle attack* etc. [33] gives a detailed survey on the security issues in cloud and a description of the types of attacks possible in a Cloud Data mining environments with their impact and possible solution to some of them. According to [19] data mining attacks in cloud falls in three classes: network-level, application level and virtualization level. [20] discusses about the Network level attacks of the Cloud system and propose a solution for these type of attacks which is deployed on IBM SCE in the form of "Security-as-Service". This application prevents the high-level security attacks. Application level security is discussed in [18]. This discusses various issues regarding the deployment, moving a service on cloud in detail. It mainly focuses on building transparent cloud application using loosely coupled services. Virtualization is the key concept of cloud computing these days but it too act as a loophole in the security of the Cloud. [21] discusses the security of the virtual network residing in a virtual environment. They first discuss the security issues in the virtual machines and network and then propose a solution in the form of a framework to control these security issues. [22] discusses the recent advances in the cryptographic security mechanism and try to apply those in the cloud environment. But, [11] states that cryptography alone cannot prevent the attacks on the cloud mining systems and some other form of security must also be imposed. Fragmentation technique or partitioning of the database into chunks [12] is another method for security which suggests that keeping the data with different cloud service provider or nodes will prevent an adversary from having the access to complete data and thus will not be able to infer correct results. A different approach is proposed in [13] for secure mining. They employs a privacy preserving repository which with a query plan wrapper limits the task of the data sharing and the access to the shared data with the encrypted results thus, maintaining the confidentiality as well. [23] discusses the k-anonymity and k-anonymity noise taxonomy in a multi-cloud environment to perform frequent pattern mining. It proves that distributed data or a multi-cloud environment prevents the attacker from getting hold of the complete data thus cannot infer valuable information from the data. A one-time pass key mechanism [19] can be used to preserve the privacy of the user as well as the service provider. This approach is based on the terminology of the authentication of both user and the provider. [24] proposes a SCM (Secure Cloud Mining) architecture for the generation of secure forecasting reports for an organization by identifying the interesting patterns and links between variables in a multivariate database system using image based encryption for secure forecasting. The secure collaborative outsourcing of data mining is discussed in [25]. This paper proposes practical scheme as most of the schemes assumes the models to be semi-honest adversary model. It presents a case study of knn (k-nearest neighbor), SVM (Support Vector Machine) and k-means in the above mentioned outsourced collaborative environment. A lot of Privacy Preserving Data Mining (PPDM) techniques exist today. [8] gives a review about all these existing techniques and analyze the representative PPDMs. It finally concludes that most of the existing techniques are an approximation and need to be perfected further if efficiency and accuracy is required as most of the algorithms compromise one for the other and to get a balance between them more robust, dedicated and perfect PPDMs are required.

III. Proposed Approach

Let D= $\{d_1, \ldots\ldots, d_n\}$ be a multivariate database, where n is the number of attributes, which holds the user's data. The Database is horizontally partitioned and stored at two locations .i.e. Host A and Host B. Host A has $D_A = \{d_1^A, \ldots\ldots, d_n^A\}$ and Host B has $D_B = \{d_1^B, \ldots\ldots, d_n^B\}$. We want to perform data mining on the given data using k-means clustering approach while maintaining the privacy of the content at both the host and also preventing the intermediate values to be leaked to the adversary. It is desired that the hosts know their inputs, the final outputs and no intermediate values.

A. Encryption Formulas

To preserve the privacy of the data of each host and the intermediate results which are communicated to and fro we need an encryption system in which if any specific operation is performed on encrypted data or cipher text, the results generated matches the operation performed on plaintext when decrypted. This system of encryption is known as Homomorphic encryption system. For this purpose we use the Pallier cryptosystem [16] which satisfies the need of the approach. We use $E(a).E(b)=E(a+b)$ and $E(a)^b=E(a*c)$ in this approach, where E is the required encryption scheme.

B. Assumptions

- A semi-honest model of adversary is assumed by the proposed approach in which a host can reveal other host's data, if not secured, while maintaining the privacy of its own.
- This approach assumes that the data input by client is stored as chunks [12] at different locations instead of storing whole of the data centrally, as, the centrally stored data is more vulnerable to the attacker. Thus the

client's data is stored in a decentralized manner by partitioning the database horizontally. *Horizontal partitioning* is referred to the partitioning scheme where each site has different records which contain same or equal set of attributes.

C. Data Distribution

A multivariate relational database depicted as D= $\{d_1, d_2, \ldots. d_n\}$ in which Host A has $D_A=\{d_1^A, \ldots\ldots, d_1^A\}$ and Host B $D_B=\{d_1^B, \ldots\ldots, d_1^B\}$. As the database is multivariate, each data object is denoted by a vector set $d_i=x_{i,1}, \ldots\ldots, x_{i,m}$ where m is the number of attributes. Now, let Host A have a set of private clustering centers $H_1^A, H_2^A, \ldots\ldots.., H_K^A$ while Host B has $H_1^B, H_2^B, \ldots\ldots.., H_K^B$ and $(C_1,C_2,\ldots\ldots\ldots,C_k)=\{ H_1^A + H_1^B, \ldots\ldots, H_k^A + H_k^B \}$ as the joint cluster centers[12]. Here, k is the number of clusters.

D. Proposed Algorithm

Notations: C_i represents the combined clustering centers which is the sum of Host A and Host B's share i.e. H^A and H^B respectively where $C_i= H^A+ H^B$.

Input: 1) Database D_A and D_B belonging to Host A and Host B respectively having n data objects.

2) 'k' which is the total number of clusters.

Output: The k cluster which is the combination of D_A and D_B or D.

1) Each party performs Data Normalization on local data.

2) Host A and Host B select their respective k cluster centers $H_1^A, H_2^A, \ldots\ldots.., H_K^A$ and $H_1^B, H_2^B, \ldots\ldots.., H_K^B$(locally) randomly.

$(C_1,C_2,\ldots\ldots\ldots,C_k)=\{ H_1^A + H_1^B, \ldots\ldots, H_k^A + H_k^B \}$

3) Calculate or perform local k-means for Host A and Host B.

4) Save the cluster centers $H_j^{A,i}$, $H_j^{B,i}$.

5) Perform the secure cluster updation and reassign the data objects to their closest clusters locally

6) Save $H_j^{A,i+1}$, $H_j^{B,i+1}$. if the difference between the previous cluster center and the current one is less than or equal to threshold value then stop the iteration else repeat step 4 onwards.

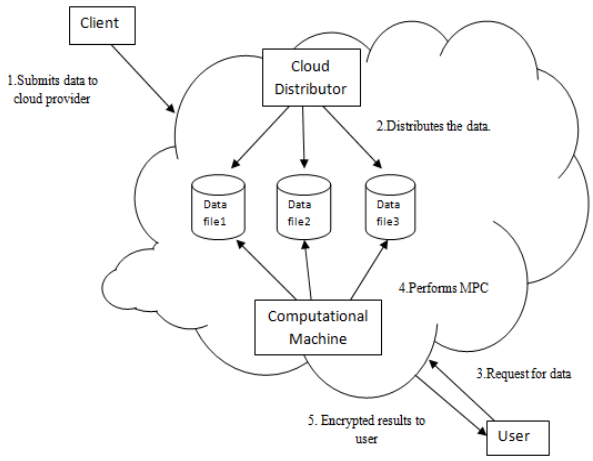


Figure 1. Overview of the Proposed Approach

IV. DETAILED APPROACH

The proposed approach uses the public key cryptosystems where M is the message or the plain text which is to be encrypted. The system can be divided into 3 parts (K,E,D):

- A pair of public and private key (l_k,p_k) is generated.
- A ciphertext or encrypted message $c=E_{lk}(m,r)$ is obtained where m € M and r is a random value.
- Decryption $D_{pk}(c)=m$ is used to obtain plain text again.

A. Private Data Normalization

A standard Xml document is used to submit the data so that a data standard is maintained. But as we are dealing with multivariate database, i.e. a multi-attribute database, the value of variable is obtained as a sum of different attributes. Thus, the probability of some variables having large values is high, which can dominate the entire metric. Thus, a normalization method is used to standardize the multi-attribute data, using private mean computation of the data objects.

Let Host A has $d_A=\sum_{i=1}^{n} d_i^A$ with n data entries

And Host B has $d_B=\sum_{i=1}^{m} d_i^B$ with m data entries

Then mean $M = \frac{d^A + d^B}{n+m}$

This mean is generated using Pallier Homomorphic cryptosystems so it also cannot be intercepted by the adversary. Now, the data is standardized locally using the above mean value as

$x_{i=}$ x-M for all data objects x_i

B. Distance measuring and updation of clusters

After the standardization of the data a local k-means is performed by all host on their respective datasets and initializes the cluster center for each attribute and assign data objects to the nearest cluster center using Euclidean or

Manhattan distance which can be chosen according to the application or database, i.e. h_1^A,……,h_k^A for Host A and h_1^B,……,h_k^B for Host B. As these cluster centers are calculated locally there is no need of any security protocol but in the next step of updating the cluster centers, joint centers are to be found which needs to be calculated privately.

Cluster Updation: for every data object's values in the j^{th} attribute in i^{th} cluster, calculate sum as

$S_j^A = C_{i,j} * n_j$ where, n_j is number of data objects for j^{th} cluster

$S_j^B = C_{i,j} * m_j$ where, m_j is number of data objects for j^{th} cluster

Now, new i^{th} cluster center for j^{th} attribute is

$$C_{i,j} = {S_j^A + S_j^B}/{n_j + m_j}$$

Pallier Homomorphic cryptosystem [16] is used to do the above computations privately as: Host A, B and Third Party randomly generates a pair of public/private keys(l_R, p_R). Host A and B encrypts their sum value with Third Party's public key and send it to Third Party along with their Public keys. As at the end of an iteration the local cluster centers are combined to get a global cluster center which is used by next local iteration, the correctness of algorithm stands true even in the distributed environment.

C. Iteration Stopping Criteria

As it is know that k-means is iterative in nature, so there must be a criteria which when met stops the iterations. This iteration stopping criteria is reached when output requirement are satisfied. For k-means this criteria is that the Euclidean distance between two consecutive cluster calculations is less then □ (threshold value).i.e. Dist (C_j, C_{j+1})=Dist($H_j^{A,l+1} + H_j^{B,l+1}, H_j^{A,l} + H_j^{B,l}$) < □ or ($H_j^{A,l} + H_j^{B,l}$)-($H_j^{A,l+1} + H_j^{B,l+1}$) < □. To check this Host A computes Enc ($H_j^{A,l} - H_j^{A,l+1}$) and host B Enc ($H_j^{B,l} - H_j^{B,l+1}$) locally with third party's public key. Then third party do multiplication of intermediate encrypted values and HOST A and B decrypt with their private key as follows:

T = Dec [Enc ($H_j^{A,l} - H_j^{A,l+1}$). Enc ($H_j^{B,l} - H_j^{B,l+1}$)]

If T< □, then the desired output is reached and the iterations can be stopped.

V. EXPERIMENTAL SETUP

1. **Tools:**
   - **Hadoop** [27] - Hadoop is a Java framework that runs applications on large clusters of commodity hardware and comprise of features like Google File System (GFS) and the Map Reduce computing prototype. Hadoop's HDFS is distributed file system that is extremely fault-tolerant and, is designed to be mounted on low-cost hardware. It is most suitable for applications with large datasets and has a high throughput access to the data being used by application.
   - **Mahout** [26] - Mahout is a machine learning library provided by Mahout and is open source. Currently it primarily implements *recommender System*, *clustering*, and *classification algorithms*. It's also provides scalability across machines. It can be the machine learning tool for the processing of collection of large data, which may be too large for a single machine. Mahout should be run on top of Hadoop when a large amount of data is to be processed.
2. **Parameters Used:**
   - k –no of the clusters. Default is 6 clusters.
   - x – No. of iterations which is taken to be 10.
   - dm – distance measure used is Cosine Distance.
   - cd – convergence delta or threshold value which is taken as 0.5.

4.7.2 Testbed

Dataset used: Synthetic_control data, which is control charts exhibiting the time series comprising of 6 different classes, from UCI Machine learning repository is used [28].

- 600 records are there with 60 attributes per record.

Technology used:

- Linux 12.04, 64-bit – 40GB hardisk,1.5 GB RAM
- Jdk 1.7.0_60
- Hadoop-1.2.1
- Apache maven-3.2.1
- Mahout-0.9

VI. RESULTS AND ANALYSIS

A. Evaluation Parameters

1. Correctness

Correctness refers to the validity of the final results obtained or the outcome of the experiments performed using the proposed approach, on the same hardware and software platform as compared to the original or base approach. The correctness is checked by comparing the deviation of the results from the anticipated results.

2. Security

This parameter evaluates the proposed algorithm in terms of security i.e. the capability of the algorithm to prevent the attackers, with malicious intent, to gain access to the confidential user data and valuable information inferred from the raw data.

B. Results

- The proposed approach performs k-means clustering on a dataset which is horizontally partitioned and stored on two different locations. The approach first run locally then performs a joint computation on encrypted intermediate results so as to obtain complete result. It was observed that running secure k-

means on the partitioned data with same parameters and computation environment as the original single party k-means, produced the same end results and same inference, thus, validating the correctness of the proposed approach.

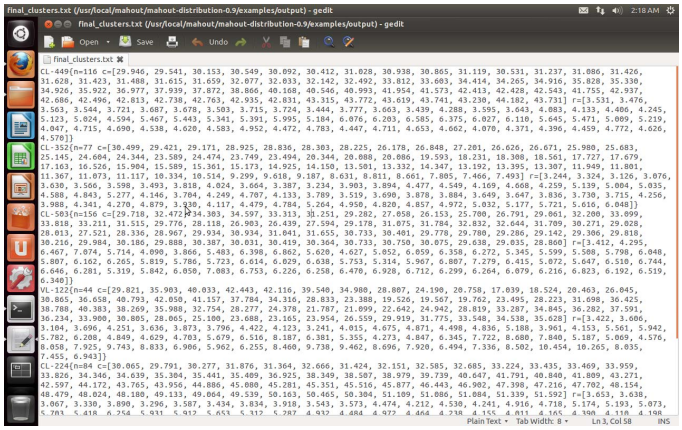


Figure 5.1: Final Clusters on Decentralized data

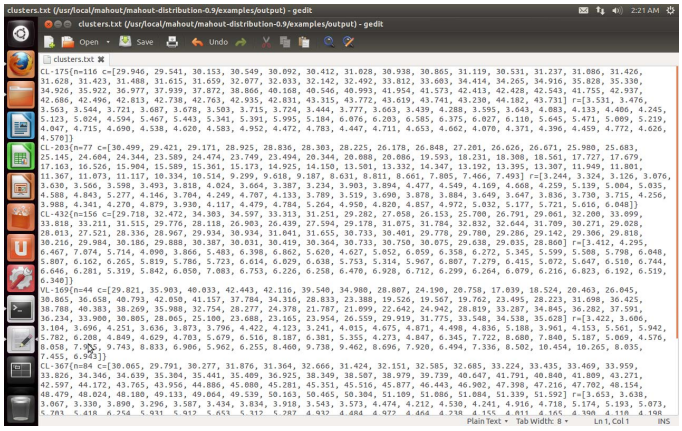


Figure 5.2: Final Clusters on Centralized data

The above figures show the correctness of the proposed algorithm. It can be seen that the final cluster centers obtained by the merging of the clusters and the clustered points obtained in the final iteration of the two-party k-mean computation is similar to the cluster center obtained by the running of k-means algorithm single time. The correctness can be further seen as both the algorithms were run on same environment and platforms with same hardware and software configuration.

Thus, it is proved that the algorithm maintains the *correctness and validity* of the final result and thus can be applied to all situations where a single party k-means can be used.

- Coming to the *security* issue we know that the model uses a partitioned approach to store the large dataset .i.e. the dataset is fragmented horizontally with a certain number of records with n attributes stored on Host A and the other set of record on Host B. Thus, *fragmentation* is the first step towards the security against data mining based attacks as the intruder which otherwise could, after getting an unauthorized access or entry to the data storage point, easily use the cheap and simple data mining techniques to extract valuable information from the data. But, as the data is fragmented and kept in chunks at different locations getting the correct information from the incomplete data becomes impossible thus fending off the attack by the adversary.

  Secondly, the assumed model is that of a *semi-honest adversary* .i.e. participants try to leak the data of one another while maintaining their privacy. This approach deals with this threat as the intermediate results of both the party goes to a third party, and that too in an encrypted form, and it performs the computation on the encrypted data and returns the encrypted results to each party. Thus, each party only knows their intermediate values and the final value but not the data of the other party.

  Lastly, as the data goes to the third party encrypted with a key, if an intruder is able to pick the data in the transition he/she will not be able to decipher the encrypted data to get the original values and to simulate the approach with those values. This prevents Sniffing attack on the data-in transit.

VII. CONCLUSION

Security and privacy is the major issue concerning the clients as well as the providers of cloud services as a lot of confidential and sensitive data is stored in cloud which can provide valuable information to an attacker. This paper proposes a method to solve the privacy issues of the cloud. It assumes that the user data is distributed on two hosts and performs a combined k-means clustering using the Pallier Homomorphic encryption system for security purpose so as to prevent any interpretation of intermediate results by an attacker. The proposed approach can further be extended by adding a digital signature or hashing technique to authenticate the third party so as to prevent an adversary from posing as the third party to host's. Also it can be generalized or extended to more number of hosts if required.

REFERENCES

[1] M. Brantner, D. Florescu, D. Graf, D. Kossmann, and T. Kraska, "Building a database on S3." In Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp. 251-264. ACM, 2008.

[2] J. Carolan , S. Gaede, J. Baty, G. Brunette, A. Licht, J. Remmell, L. Tucker, and J. Waise, "Introduction to cloud computing architecture." White Paper, 1st edn. Sun Micro Systems Inc (2009).

[3] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica. "Above the clouds: A Berkeley view of cloud computing." Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS 28 (2009): 13.

[4] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina. "Controlling data in the cloud: outsourcing computation without outsourcing control." In Proceedings of the 2009 ACM workshop on Cloud computing security, pp. 85-90. ACM, 2009.

[5] D. J. Solove. "I've got nothing to hide and other misunderstandings of privacy," San Diego L. Rev. 44 (2007): 745.

[6] P. K. Rexer, "Data miner survey highlights the views of 735 dataminers" 2010.

[7] C. Su, F. Bao, J. Zhou, T. Takagi, and K. Sakurai, "Privacy-preserving two-party k-means clustering via secure approximation." InAdvanced Information Networking and Applications Workshops, 2007, AINAW'07. 21st International Conference on, vol. 1, pp. 385-391. IEEE, 2007.

[8] Md. Riyazuddin , Dr.V.V.S.S.S.Balaram , Md.Afroze , Md.JaffarSadiq , M.D.Zuber. "An Empirical Study on Privacy Preserving Data Mining". International Journal of Engineering Trends and Technology (IJETT). V3(6):687-693 Nov-Dec 2012. ISSN:2231-5381

[9] K. Che, and L. Liu, "A random rotation perturbation approach to privacy preserving data classification." (2005).

[10] A. Inan, M. Kantarcioglu, and E. Bertino. "Using anonymized data for classification." In Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on, pp. 429-440. IEEE, 2009.

[11] M. V. Dijk, and A. Juels. "On the Impossibility of Cryptography Alone for Privacy-Preserving Cloud Computing." IACR Cryptology ePrint Archive 2010 (2010): 305.

[12] H. Dev, T. Sen, M. Basak, and M. F. Ali. "An Approach to Protect the Privacy of Cloud Data from Data Mining Based Attacks." In High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:, pp. 1106-1115. IEEE, 2012.

[13] R.Mishra, S. K. Dash, D. P. Mishra, and A. Tripathy, "A privacy preserving repository for securing data across the cloud." InElectronics Computer Technology (ICECT), 2011 3rd International Conference on, vol. 5, pp. 6-10. IEEE, 2011.

[14] M. D. Singh, P. R. Krishna, and A. Saxena. "A cryptography based privacy preserving solution to mine cloud data." In Proceedings of the Third Annual ACM Bangalore Conference, pp. 14. ACM, 2010.

[15] S. Pearson. "Taking account of privacy when designing cloud computing services." In Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing, pp. 44-52. IEEE Computer Society, 2009.

[16] P. Paillier. "Public-key cryptosystems based on composite degree residuosity classes." In Advances in cryptology—EUROCRYPT'99, pp. 223-238. Springer Berlin Heidelberg, 1999.

[17] K. P. Lin, and M. S. Chen. "Privacy-preserving outsourcing support vector machines with random transformation." In Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 363-372. ACM, 2010.

[18] R. Bhadauria, and S. Sanyal. "Survey on security issues in cloud computing and associated mitigation techniques." arXiv preprint arXiv:1204.0764, 2012.

[19] R. Bhadauria, R. Borgohain, A. Biswas and S. Sanyal. "Secure Authentication of Cloud Data Mining API " arXiv preprint arXiv:1204.0764, 2012.

[20] K. Beaty, A. Kundu, V. Naik, and A. Acharya. "Network-level Access Control Management for the Cloud." 2013 IEEE International Conference on Cloud Engineering (IC2E), IEEE, pp. 98-107, 2013.

[21] H. Wu, Y. Ding, C. Winer, and L. Yao. "Network security for virtual machine in cloud computing." 2010 5th International Conference on Computer Sciences and Convergence Information Technology (ICCIT), IEEE, pp. 18-21, 2010.

[22] I. Agudo, D. Nuñez, G. Giammatteo, P. Rizomiliotis, and C. Lambrinoudakis. "Cryptography goes to the cloud." Data Management, and Applications in Secure and Trust Computing, Springer Berlin Heidelberg, pp. 190-197, 2011.

[23] C. Tai, J. Huang, and M. Chung. "Privacy Preserving Frequent Pattern Mining on Multi-cloud Environment." 2013 International Symposium on Biometrics and Security Technologies (ISBAST), IEEE, pp. 235-240, 2013.

[24] ASA. Ansari, and KK. Devadkar. "Secure cloud mining." 2012 IEEE International Conference on Computational Intelligence & Computing Research (ICCIC),IEEE, pp. 1-4, 2012.

[25] Q. Lu, Y. Xiong, X. Gong, and W. Huang. "Secure collaborative outsourced data mining with multi-owner in cloud computing." 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) , IEEE, pp. 100-108, 2012.

[26] S. Owen, A. Robin, T. Dunning, and E. Friedman. Mahout in Action. Manning Publications, 2012.

[27] http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-single-node-cluster

[28] http://archive.ics.uci.edu/ml/databases

Towards realizing the Secured Multilateral Co-operative Computing Architectural framework

Manu A R
IEEE-ACM Member,
PESIT, Bangalore, India
manu.a.ravi@gmail.com
manu_ar@nitk.ac.in

V K Agrawal, *PhD*
Senior Member IEEE
Director and Professor, CORI
Former Scientist ISRO, PESIT,
Bangalore, India vk.agrawal@pes.edu

K N Balasubramanya Murthy, *PhD*
Vice Chancellor, PES University
Bangalore, India
vice.chancellor@pes.edu

Manoj Kumar M
Research Scholar
JU, Bangalore
dbamanoj@gmail.com

***Abstract*—*Innovative approaches for securing the computing systems have intense inference for our understanding of technological, societal, economical, and political phenomena in making guiding principles, policies, rules and security implementations. The model presented here is based on cooperative and collaborative multilateral relationship among the shared business community partners. The sole responsibility for the securing and maintenance of their own virtualized dedicated boxes with jointly hosted data centers distributed geographically is equally, vested with service consumers and vendors. Inspired by multilateral techniques used in army and health care applications, this model is a conceptual and empirical tool aimed, rather depicting a particular set of observed situations or making predictions. It is aimed is to develop our understanding of the fundamental mechanisms driving the security implementations of existing methods and provide the MCF - multilateral collaborative co-operative framework. This MCF demonstrates an interrelated multilayered virtualized architectural framework for computing utility using virtualization platform. We try to demonstrate qualitatively and empirically the proposed architecture works well for a wide range of workloads and devices belonging to multi tenants with varied security needs. This work is compared with currently existing virtualization platform security framework and avail the novelty of the proposed ontology and framework.*

***Index Terms*— Multilateral Cooperative computing, Virtualization, VMM, utility computing, secured grid computing, cloud computing**

1. Introduction

C*loud computing* is a computing system which delivers numerous hosted services and resources over the Internet. This computing structure aggregates virtually interconnected computing facilities shared cooperatively over web between service providers and consumers [79-80]. This computing facility is shared among multiple tenants and stakeholders located at multiple countries across the geography [1-6]. Cloud customers perceive difficulty owing to faster growth rate of the complex data centers. Thus giving rise to the concerns of clients losing control over their data, portability, security, trust and privacy issues [69]. Also non-standards in security offering from the vendors, legacy and legal jurisdiction issues [69]. The security offerings from the cloud vendors find difficulty to meet the various demands for security needs of multiple customers owing to complexity of the cloud model. At present the cloud security is offered through SLA's (service level agreements), policy, rules guidelines and trust based on restricted security offerings of the cloud vendors.

In reality the sole responsibility of security and privacy controlling is equally vested with service consumers and vendors. To overcome the security drawbacks (listed in ref [69] of cloud computing system, we propose multilateral co-operative computing framework (MCF). This MCF is built on top of existing security standards, assists in automating the security management process using an interrelated layered, secured virtual architectural framework. This MCF provides the flexibility according to changing scenario of the threats with attack complexity and vulnerability using security requirements of various stakeholders. The proposed multilateral co-operative computing architecture consists of virtualized multilevel features with multi-factor authentication [82], based on multilateral co-operative and collaboration, using virtual machines (VM). The contribution of this paper is introduction of layered architectural framework helps in security implementation at each layer using MCF. We cluster the security requirements (SR) using Bell-lapadula, Biba, BMA, Clark-Wilson models [74-75] and create compartmental mechanism that provide the security decision support to the de-centralized - distributed VM's at various data centers across the geography.

The methodology involves capturing the requirements and performing feasibility study using qualitative and quantitative methods for scrutiny of SR (Security Requirement) for each layer. Also assess the risk impact factor with clustering and grouping of the requirements. Then we create the compartments of requirements of each layer. In addition we use the Chinese wall security system [74-75] for each of the compartments. Subsequently implement, and execute the requirement of each tenant. Then we use MCF for joint monitoring, maintain and enhance the security of the resources and services. The next section of this paper presents basic requirements, assumptions and background for proposing MCF. Section-3 presents MCF computing system layered architecture. Section 4 presents design of MCF, algorithms and mathematical formulations. Section-5 presents comparison of related work, with the merits and demerits of our proposed work. Finally the scope and future directions are presented in section 6.

978-1-4799-6141-2/14/$31.00

2. Basic needs and background for proposing MCF

The main objective of proposing generalized MCF is to provide flexibility for each stakeholder to tackle cloud security issues and attacks by implementing their varied SR's. At present the existing cloud security measures include - strong physical and virtual security monitoring using CCTV line video streams 24X7 across the datacenters, along with monitoring the system logs and user traffic, by revising it periodically. Defense is also provided by performing mandatory regular internal and external auditing and accounting of the systems. In addition, security is provided by using multifactor user authentication with regular password change policy. As per industry standards service management and security framework for multi-tier mapping with 3– tier architecture for specific IP addresses. In addition, defense mechanism is provided using encrypted form for data migration over Virtual Private Network. When the data is at rest, data are stored in encrypted form. Here, security is provided by deploying Intrusion Detection and Prevention Systems, animating malware with strong firewall and policing the system. In addition, security measures also includes verification of the employees details working in each datacenters, installing rogue system detection, appliance blocking, and information condition base-lining [69].The existing utility based collaborative computing offers amortizing local geographic region, where the costs for power, cooling, water, manual labor, property costs and taxes are geographically variable [1-60]. The existing method to secure VM by physically shipping VM's over fiber optic cables or through ships or overnight delivery services , using cargo shipping fleet/air service. To avoid any physical attack this should be done within stipulated time found in ref [6-9].

In order to present the advance security features for cloud computing services. We need to consider domestic and global technique in security requirements for cloud computing. It also demands to design of secured protocol, with security visualization, information isolation among the competitors. Ref [2] discusses the need to know the data, software, activity patterns of sharing of resources among multiple parties. VM migration found in [7] helps in fixing legality and legal liability issues. VM Consolidation with multiple applications and sharing of multiple distributed resources is provided in [8]. Ref [17] provides the layered architectural model is taken as the base footprint of our model is shown in fig 1 [17]. The reference [18- 21] provides the insight to create physical server farms distributed geographically named as distributed data centers. [22- 23] suggest regular monitoring, fault detection, error acceptance and mechanical recovery, meant to be integral to the system architectures. Reference [51- 55] suggests virtualization, migration, consolidation of VM's to solve the problem of server idleness and low utilization.

The proposed MCF uses all the existing security measures along with new security requirements of multiple tenants. In actuality it is difficult to afford and implement the security needs of each stakeholder, this impact the service performance and business needs using multi-tenancy quotas. Source code reviews of virtualization, visualization of access control through general and special managing, monitoring, and analyzing the logs with analog management, and knowledge of VM's, data centers helps in dispelling the security concerns. Based on [17], we present improved version of multilevel security life cycle shown in fig 2. VM consolidation is categorized into semi static consolidation, static consolidation, and dynamic consolidation grouping is shown in fig 3 [51-55]. Our proposed theoretical framework discusses benefits, norms, strategies, structures, and sources of legitimacy. MCF helps for collective innovation process and identifies the interdependence among the service utilities. It also assists to establish the de-facto standards to close jurisdictional gaps to align the privacy, expertise, experience and knowledge of tackling the security threats and attacks. By establishing the multilateral relationship among the various stakeholders we combine and cluster the idle VM's in the data centers through MCF to create the compartments using Bell-la-pa-du-la model [74-75]. This helps in better utilization efficiency of resources and also improves the security of VM's by securely archiving the VM's.

3. Layered Design of MCF

MCF is based on multi-layered, multilateral secured hybrid cooperative, collaborative, multi-community based computing framework. MCF is designed with authenticated access policy, plan, principles, preparation, procedures, practices, action, ethics, and doctrine among the multiple countries across the world irrespective of provincial border. With permissible right to use computer resource or applications, or even to physical areas and buildings located geographically across the world. The layered architecture is responsible for underlying management, provisioning of hardware components, memory, and process management. This architecture separates functionality into hierarchical layers, provides the easier way of implementation of multilayer security. It also helps in designing security at each layer from starting to end. In addition, it allows intended for easier future security enhancements with minimal and isolated controls structure with relevant security features to each layer. The computing utility is accessed using specially designed inexpensive wireless hand held mobile access devices and thin clients with low cost wireless mobile or existing gadgets, and widgets appliances.

End user uses excellent secure standardized unique graphical user interface (GUI) and multimedia support. Subscribers are connected to Internet through web to solve their computing problems cooperatively.

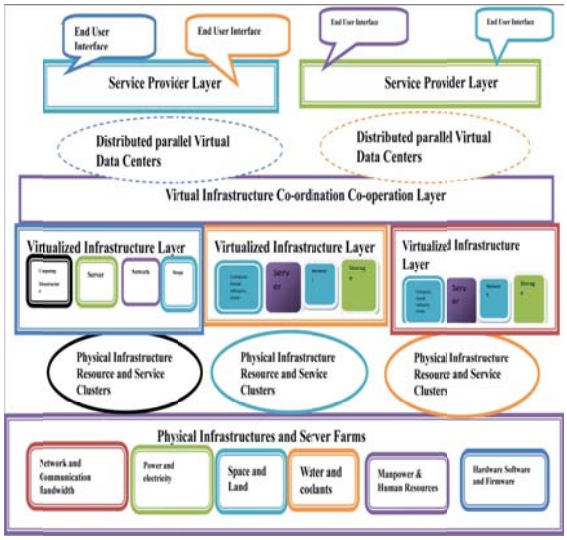


Fig 1 - [17] based multilateral, distributed co-operative model, with layered Virtual Organization

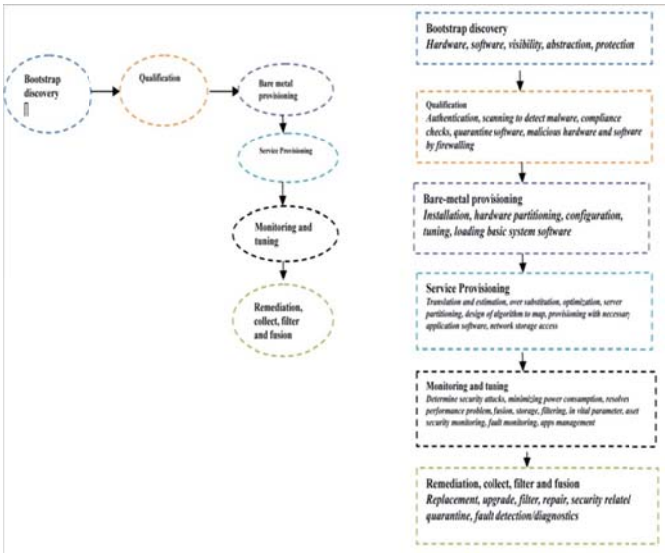


Fig 2 – improved version of life cycle of MCF

By using either wired or wireless system to a service broker's (a computer with appropriate software). The request is processed using service request language and voice (and if needed video) to provide access to consultants from service providers. Clustered community of multilateral cooperating service providers maintains this computing utility. Vendors send request to the provider through broker to provide the required service. (It is anticipated that brokers advertise the services provided by several multilaterally co-operating vendors). The broker quotes the cost and time estimates for the service requested by a subscriber after negotiating with a resource manager, the broker is connected to it. It provides quality of services (QOS), on time backup and disaster recovery plans, business continuity plans, integrity and availability of routine service assistance and consultancy help (free or on payment) to patrons opted services. Also with service guarantee provided to subscribers by providers as per SLA. The VMM resource manager (which is a computer with appropriate software) maintains a table with the status of all computing servers, application servers and file servers connected to it. VMM schedules the service requested by the broker using a scheduling program. The resource manager also dynamically shifts the load, if any failure thereby providing fault tolerance and ensure QOS. We try to provide layered architecture diagram with improved security and usage with spawned copious novel applications. The block diagram of fig 5 describes succinctly the hardware and software systems that constitute a co-operative computing utility. The components and security measures required in the proposed layered architecture of cooperative computing is as follows: **a) Physical infrastructure/Resource layer (Fabric and Communication layer):** Here in this layer we have flexibility by adaptability to changing technologies, policies and regulations with increased spectrum. Also with power efficiency, opens a way for spectrum sharing, active infrastructure sharing leading to polarization of services. A cognitive cloned computing service is possible using this medium. Computing resources include computer server farms, coaxial fiber, wireless network devices. The components include: Hubs, repeaters, amplifiers, transceivers, multiplexers, cables, network cards, servers, hard disk, computer, monitors, networks, and network communication devices. The physical characteristics of network include connection, voltage levels and timings, software hardware and firmware etc. In addition, networking, software defined networks, Internet of Things, bandwidth, Internet, and web. Also includes computing power, storage media, clusters, services, instruments, databases, OS, software libraries, business processes, instruments, user interface, shared resources, computing infrastructure, pooling, integration, fabrication etc. b**) Network and data-link connectivity layer:** It helps in unified threat management, provides virtual private networking (VPN), and Internet protocol (IP), physical network (wired and wireless). It defines logical, virtual address specific protocol, services according to unique address. It establishes link using logical and virtual link, it works on (MAC) Media access control access method, the devices operated in this layer are router, b-router, frame relay device, ATM switch devices, bridge switch, ISDN, intelligent hub, NIC etc. it helps in determining the best route. This provides hop-to-hop collection. It uses frames, CRC, LLC, actual network interface and SAP on computer or other device, NIC's, checks initial connection setup. It keeps tracks of how many hops are active and works on unified threat management. **c) Secure transmission layer of data storage and transport layer**: It is called as access encryption host to host layer. It works on sequencing access authentication, data encryption, provides reliable services for data transfer and service access. This helps in end to end connectivity between application/service/resources and end-

to-end error recovery. It also ensures complete service flow control. d) **Clustering layer with service session and resource management layer:** Interpreting resource request from broker provides the input table of available resources to broker and standard cost estimate for clustered service type. It helps in negotiating with brokers with optimal scheduling of different requests from brokers for services on the servers. It performs job process scheduling, and monitors progress of jobs. Using check points and ensures fault tolerance limit, reschedules jobs in case of failure of the resource. It helps in disaster recovery and business continuity. The resource manager examines the request based on the current resource quota management. It also insures QOS and queries for services with standard IT resources, deployment and re-engineering, statistical and dynamical asset management. In addition, life cycle with industry tailored and customized asset management with security, privacy and trust management. e**) Resource/Services/device/instrument broker layer for computing utility:** Utility broker act as an intermediary between patrons (consumer offered services) and clustered co-operative resource/service providers. The broker layer authenticates the patron and provides the access control to the pool of resources. A broker also quotes cost for the requested services and estimated time of completion of the job request based on the information provided by the resource manager. Customers' requests are interpreted and forwarded to the resource management layer. This also negotiates with resource manager on QOS, cost and time estimates and forwards the same to the customer. This layer includes API's, hardware, software, firmware, middleware, SDK's, co-allocation CPU, peripherals, virtualization and consolidation, reduction of OPEX and CAPEX. In addition to use resources completely, up gradation, responsiveness, metering, billing, accounting, licensing, distribute, indexing, access to services and resources, platform, service cataloging, ordering, flexible pricing etc. f**) Customer Access layer (Apps and Interface Layer):** Patron access device/instrument/service using GUI. Service requests presented to the broker using a special simple command line language. The customer's use cheaper mobile, hand held devices, thin/thick clients, light weight embedded computer with wireless access to the broker's computer. It is largely distributed cooperative and collaborative utility has several brokers distributed geographically. This layer allows patrons using wireless thin clients to access the nearest broker using PUB-SUB model. It helps in design, build and troubleshooting resources, network services with QOS. All devices and resources on network speak through same language protocol/a Pub-sub agreement. User access the service/utilities through medium called as physical network. It standardizes network components to allow multiple vendors, development and help desk support. Promotes multi-vendor interoperability for the products/services or resources produced/offered/serviced to meet the same networking standards and accessed using same networks. Interface - security, database storage, file system, communication system, abstraction, optimization, visibility, automation across IT and business, pilot execution, consulting and commitment. **g) Access control and Security layer:** Process, plan, prepare, produce, secured product, access control mechanism with end to end customized tailored governance, risk management, compliance for business, threat management, disaster recovery and backup with rapid adoption, faster response to risks. It also includes helpdesk, opportunity, authorization, authentication, monitoring, firewalls, VPN's, auditing, accounting, critical analysis, conflict analysis, portfolio and threat analysis. **i) Energy Efficiency layer:** Greener energy, greener compliance, measures to use a lesser amount of water and energy, use solar energy, low radiation, reduction in heat, environment friendly. This layer address energy issues, compliance for business, sustainability and renewable challenges and opportunities.

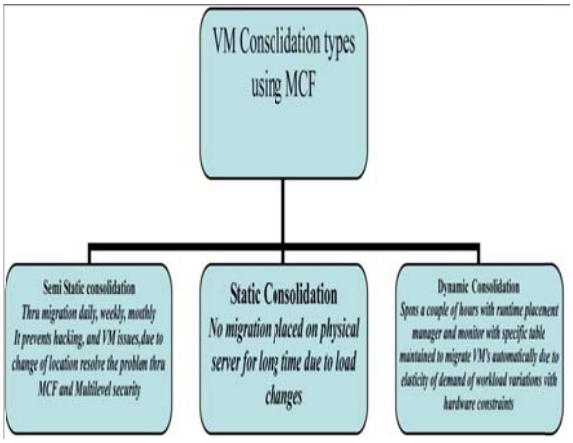


Fig 3 depicting types of VM consolidation

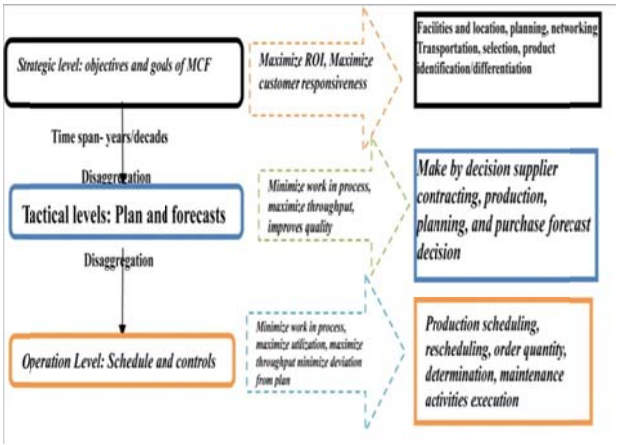


Fig 4 showing steps to implement MCF

3.1 Methodology for secured MCF

MCF mainly focuses on security and in general load balancing by sharing the load over different data-centers to accommodate the VM's, services or resources, with environmental friendly maintenance. The conceptual layered architecture helps in realizing importance of security needs of each layer. Security policy document specification captured from stake holders, helps to maintain the confidentiality, integrity, continuity using the SR of multiple stakeholders. We use *Bell-lapdula* (formal) model [74-75] to maintain the confidentiality and continuity among the various stakeholders. To maintain the integrity of SR using the clustered services and resources we form the compartments preserving integrity. We use *Biba model*, *Clark-Wilson (informal) model, Harrison Ullmann Model* [74-75] for tight security in compartments. SR may require dynamic changes we use *Chinese wall Model [74-75]* to accommodate the changes and safeguard the clustered services in the created compartment– (e.g. access rights).These models are represented in mathematics, and analytical ideas with empirical and conceptual ideas in ref [74-75], most of the times ideas/mechanisms later mapped to structure specification.

In some forms lattice is mathematical construct built upon the notion of group with set of elements using clustered SR's [74-75]. This includes partial ordering relation between the set of SR with unique least upper bound and greater lower bound feasible to implement tight security features for compartmentalized requirements. A security lattice model combines multilevel and multilateral security labels with set of categories based on state machine models. *Bell-la-pa-dula* (BL) model (inference flow model from one level to other) [74-75] is mathematical model designed for confidentiality with multilevel security policy- with different clearances, using no read up at higher security level, no write down rule at lower security level, with permitted read/write only at same platform security level. We also use *Biba integrity model* [74-75] – to maintain the integrity of various SR of multiple tenants/stakeholders in the layered architecture. It sets up no read from lower level, in addition, to write up in the higher level, this helps in setting up discrete compartments. Where as in *Clark-Wilson model* [74-75] address the integrity of security implementations, with constraints for separated compartment clusters with one subject/compartment need highly protected cluster item and another subset with unconstrained SR. This model helps in maintaining the internal and external consistency, with separation of duties prevents authorized users from making the improper modifications. This model allows the information flow from up and down, to and fro, cross domain. Check audit logs, tracking the shared control mechanisms operating in parallelism in distributed datacenters. By using the information model – *BL and Biba* model to set individual discrete MCF compartment with layered cluster of various SR implementation based on common demand of stakeholders community.

Compartment is created based on classification and clustering of SR's using negotiation. We use Brewer-Nash Chinese wall model for dynamic changes in SR based on user's present and previous experience due to security attack/threat faced. It helps to resolve conflicts between the users, prevents the information flow that creates the conflict among the clustered compartments. Compartment with demilitarized zones for placement of VM's prevent the attacks with wide range of tactics and collective action with perimeter defense.

Our MCF is designed to maintain the confidentiality, integrity, availability, security, continuity of services across the globe. MCF should define security relevant parts of the arrangement through layered architecture. Create the relation and interaction between compartments defined. MCF should evaluate assured levels of SR selected through negotiation. Consolidation of secure methods and different ideologies related to security evolved over time, define how to capture the SR's of various stakeholders, cast vote, rate for negotiated SR's through top, bottom and cross communications among the tenants. Finally MCF helps to evaluate and create trust between vendor and customer group among stake holders. SR captured from various stakeholders must be clustered based on explicit, well defined enforced mechanism. Document the test, design, guidelines, manuals, specification documents of associated SR and implement the SR in efficient way throughout life time continuously at different situations. In addition, compartmentalizing the system helps the system breaking into units, keep the complex system simple, implement the distributed system simple, assures the privacy preserved, promotes the use of community resources, helps in complete mediation among each layered object is checked. MCF helps to open for scrutiny by the community of stakeholders who share the responsibility of maintaining the security of the system. It helps in implementing SR of all community users using joint management, with separation of privilege through compartment. This helps to build the trust base among the community of stakeholders. The layered clustered architecture provides channel to check the integrity, accountability of the computing system distributed across the geography. Establishing the domain based compartment helps the components to set rules and practices that dictate how sensitive information and resources are managed, protected and distributed. This helps to define the security perimeter and provides the well defined interface between the layers.

MCF helps to meet security claims of vendor and consumers and meet user specific requirements, helps to develop security standards among multiple countries through multilateral relationship, negotiated with assurance level. MCF designed on distributive autonomous adaptive basis with fast, spontaneous planning of resources when the system is overloaded, share the load over identified datacenters, with distributed scheduling. Dispatch of distributed rescue units, co-allocation of rescue units, helps

to deal multilevel, complexity, diversity and ubiquity issues. Compartment is created based on homogeneous components and security needs, with distributed, decentralized multiple technologies at multiple place with global performance. MCF helps in rethinking security of communications networking and computing paradigm. Sets up joint monitoring, joint co-operation behavior in the face competition. MCF involves overall cross domain activities and ensures a common approach for the implementation of cross domain capabilities with mission partners.

3.2 Algorithm

The proposed architecture is designed in 6 phases:

Phase-1: Defining the Security Requirement

a) Capture the security requirements from all the customers spread across the geography. b) Cluster and pool all the requirements based on the need and threat priority faced by the users. c) Identify the conflicts and make feasibility study. d) Outline the scheme. e) Put for rating and voting through the communication and form the compartment based on clustering. f) Categorize the hardware and software required to implement the security requirement. g) Forecast the completion of the study.

Phase-2: Discover and define the requirement and negotiate the needs between the customers

Define the boundary, limits and procedures of the system, can be realized and complemented that do not affect the computing system performance, simplifying the large-scale system in to small one, omitting the unwanted details, aggregation, segregation, abstraction of the security requirement and substitution with the existing security system. Form the compartments based on negotiation and align.

Phase 3: Data Collection and Preparation

a) Identify the complexity level of the security required, conflicts in the need to complete the feasibility study, b) Categorize and cluster the sources of input and security requirements from the customer based on priority and impact of security/risk/attack/threat/issue, c) Design and document the techniques used to collect requirements, d) Verify the goodness of fit, e) Negotiate the need and level for feasible implementation and impact on the system.

Phase 4: **Verification and Validation**

Building and running simulation of security models of the identified need through voting and rating on the clustered pool of the security requirements stored in the factory or plant. **i.** Verify and refine the model and communicate the same to all customers spread throughout geography**. ii.** Validation: Verify the simulated models, methods used that behave as intended or not and validate adequately with representation of the real system, using walk through. Test run with different threats faced on the history and past experience, using interactive debugging, Animating the model and observe behavior and verify the model as correct structure and has important features of real system with levels of the security needed at each level of the MCF.

Phase 5: Output analysis

Scenario analysis output simulation model, resource utilization, and flow times with tight security implementation.

Phase 6: Documentation of output and communication

Calculate queue lengths confidence intervals, communicate the results of simulation study and make recommendation to implement the SR belonging to all patrons.

3.2.1 Description of the phases

The SR is captured from various stakeholders/multi tenants, then cluster the SR based on frequency/consequences/toleration limit. Clustering is done with reliable separation and organization decomposition into multiple independent levels of clusters on multilayer into security critical and non critical functionality SR. Form the multilevel workstation (WS) of the data-centers on interactive basis, handle multiple concurrently separated services. This WS of VM's design involves the concept of distributed infrastructures and shared equal responsibility to stakeholders across the geography. This involves the multilevel networking, with secure GUI ensures unambiguous co-relation of user interaction with well defined services. Multiple VM's in distributed data-centers are connected using dynamic configuring process along with security association.

Modules followed in the model: a) Build: Define objective, plan, report, and service. The security requirement capturing, created based on identification, rating and voting. **b) Impact study**, feasibility study of security requirement on the performance of the system. Store the requirement in factory/registry/plant/database. **c) Assign:** values of security type and processing issues categorize and cluster. **d) Queue:** the requirement. **e) Confiscate**: if it's affecting the utility performance. **f) Delay:** Till the voting, casting, and rating is finished. Consolidate, migrate, allocate, provision, cluster, negotiate and multiplex VM's for security reasons to minimize the server idleness, security requirement/control selection, set and generate baseline to tailor the solution. **g) Release:** if not feasible security requirement. **h) Compute and compare:** contrast the selected requirement from the factory with the existing security measure. **i) Count:** number of requirement in the factory, set the status and priority. **j) Tailor and Execute:** Security control implementation. **k) Evaluate and assess:** monitor the system performance, decide and change if required. **l) End Dispose:** remove the old security requirement after modification and refinement of the implementation. **m) Replicate**: communicate to all patrons after modification. Fig 6 shows the steps to implement MCF. In case of conflicting demand for close co-operation with multiple compartments protected from one another, provide

entities under joint supervision, with seamless integration of existing application and business process. SR can be met cost efficiently by our high security architecture with riskless integration with extension and adaptation through negotiation, voting, polling, rating of SR is easily possible. Multilevel workstation advocating and implementing interim needs (SR's) of customers based on consensus created using reasoning, plan of action, negotiation and rating. The responsibilities among various stakeholders involve joint effort and evaluation and shared responsibility of securing the systems and finalize SR for implementation with active participation [70-74].

The negotiation is based on conceptualization, design, implementation, by harmonizing, negotiating among many nations, for community product dissemination through collateral relationship. MCF gives a single integrated consistent security infrastructure for all SR needs, using community archetypes. Multiple VM workstations connected to distributed data centers with cross domain interactions using web shield, VM searches cloned to multiple security levels using modular decomposition. At each layer we implement the policy mechanism - middleware. Design the system allowing for the flow of information from up and down, to and fro, and cross domain to realize the MCF. Establish negotiation through business lawyers, managers, network operators, legal experts of governments, NGO's. Define business, legal and technical aspects where participating parties adhere to setup an agreement [72].

.

Based on BMA model [74-75] we design the process of negotiation for MCF algorithm for the involving stakeholders and getting their consent for captured SR, based on feasibility study accomplished by the stakeholders.

1. Establish the SR for access control to the designed compartment with grouping SR through the impact study
2. Group, classify and cluster for rating, polling, casting, and voting among the community of users.
3. Set joint task force for joint monitoring and vigilance and policing using the community of users
4. Control: Held responsible among the community of users.
5. Notify and consent: the responsible must notify if any security changes to be made. Partners need to avoid conflicts of interest dealing.
6. Persistence: Keep the security update as per SR, and provide flexibility to change.
7. Attribution: Audit trial accounting of logs and user traffic maintained subject to periodical revise.
8. Information flow: should be well established with top, bottom, cross domain flow.
9. Aggregation Control: All involved entity should get special notification in changes made and appended to the existing system.
10. Establishment of trusted computing base among the community of users.
11. Provide the indoors knowledge of plans and status or stance of the competitor.
12. No other person access the data objects on the other side of the compartment wall.

MCF uses unity of purpose security goal with IT and information distributed over a vast and wide ranging enterprise. With domestic and international partners actively participate in MCF mission where the good services, resources, are shared and secured in coordinated way with welfare sustained. MCF implemented as a joint global enterprise is operated in the world with skilled user's partnership with intelligence community to do mission-critical operations. MCF operates with joint partnership and supervision channels workforce and deny adversaries and provide security that prepare for and operate through cyber attack. To counter any kind of attack and counter future attacks, help any mission to configure dynamically and automatically. It also sets the stratagem plan transition to de-centralized and distributed form with setting up joint task force of security experts for global network operations and degradation of attack [70-78]. Partitioned to establish the unified cross domain management office to synchronize and accelerate the availability of the services of all levels. In addition it helps to foster collaboration, co-ordination grouping partnership, and pilot to build an international program to aim to achieve robust machine to machine network defense capabilities throughout the communities. Create realistic and secured environment, for all critical infrastructure sectors. This protects coalition and allied operation has accelerated international co-operation with common objective to promote adoption of international standard and norms in partnership with inter-agency processed [70-78].

General Modeling:

1. The entities flowing through the utility system are the jobs (customer security requirements with identified conflicts) being machined with 3 attributes: a) **Requirements priority**, impact factor of security/attack/risk/type of threat on business, b) **Process time**: Feasibility and business impact of security measure, based on rating and voting. c) **System**: Value assigned from the appropriate uniform distribution and time frame required to implement the level of security requirement of the various customers.
2. **Arrival time**: Negotiated requirements based on priority and rating the security requirement for implementation of the job in the system and recording the flow time statistics.
3. **The agility** of security measure implemented is that stronger and complex, the more agile up to point of security measure implemented and heavier in depth ***(h)*** of the security requirement,

Mathematically:

$$Agility\ of\ security\ requirement\ \alpha\ \frac{Strength\ of\ the\ security\ requirement}{Supremacy\ of\ the\ security\ requirement}$$

Assumptions: It is related to strength of MCF, with multifactor, multilevel, multidimensional security systems. b) Requirements that are geometrically similar are clustered, and negotiated with the customers. c) Abstract, consolidate, migrate - VM's for allocation and provisioning as per the need and minimize the idleness of the servers. d) Density of security requirements are clustered with identical density.

Mathematically:

Strength of the security requirement $\alpha \quad h^3$

Supremacy of the security requirement $\alpha \quad h^2$

$$Agility\ of\ security\ requirement\ \alpha\ \frac{Strength}{Supremacy} \alpha \frac{h^2}{h^3} \alpha \frac{1}{h}$$

So the agility of security implementation increases with depth of security requirement, and measure. Lesser the depth, lesser will be the agility, greater the speed, strength and supremacy, the agility is more.

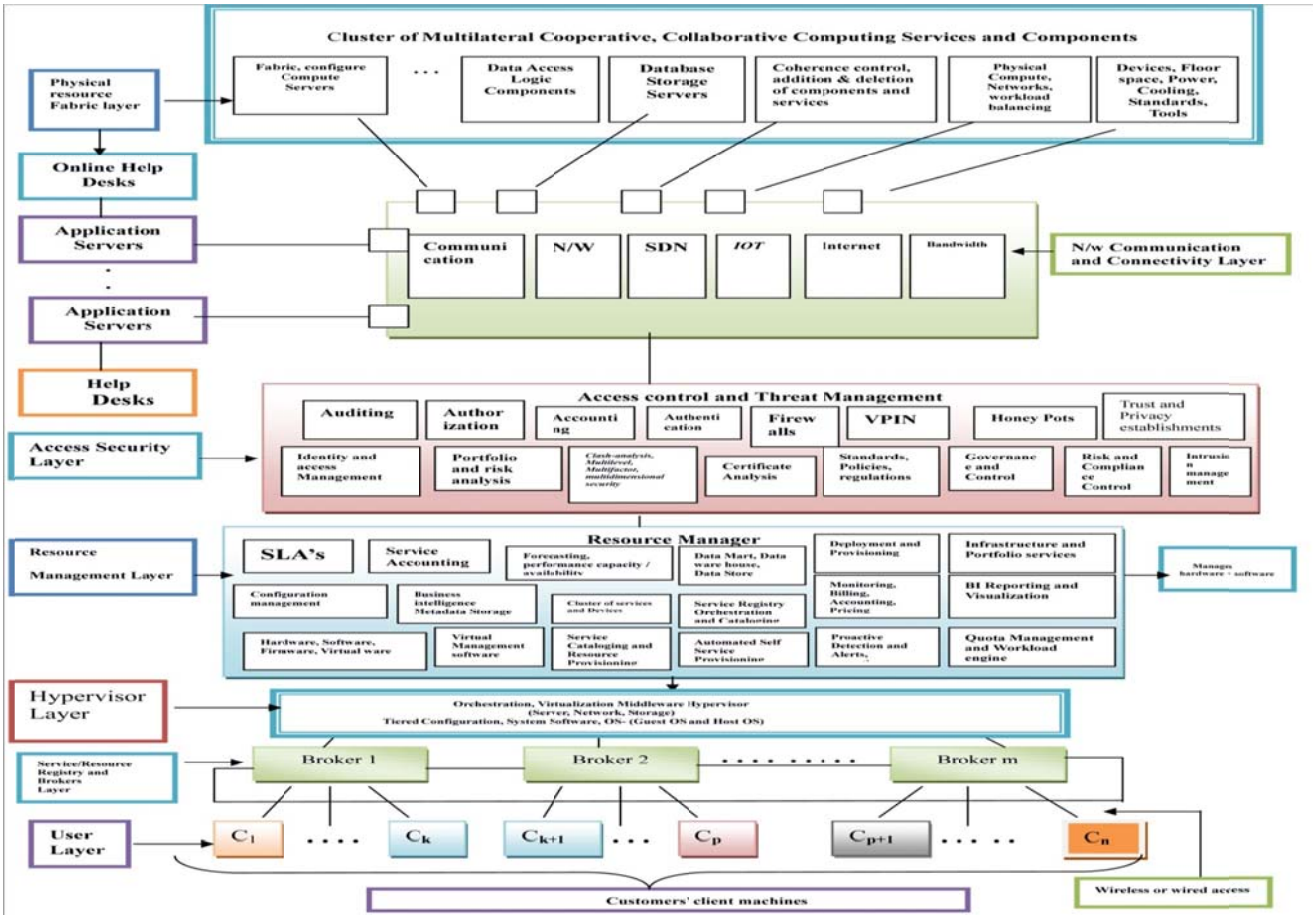


Fig 5: Block diagram of a cooperative computing utility

3.3 Mechanism to improve dynamic resource discovery and mapping using PUB-SUB model

Mechanism to discover and match shared capabilities with distributed resources and configuration algorithms offer fully distributed environment [72]. Design dynamic algorithms to map standard federation goals to specific resources, configure security requirements with varying goals and tasks. Novel resource discovery algorithm and protocols need to be designed, capable of handling number of different resource types and attributes not necessarily known at design time. Novel protocol that support fully distributed multilateral negotiation need to be devised. To achieve the process of maturity, we need holistic control systems, security standards and transparency. We use Pub-Sub model [73] [76-78] where publisher/vendor/provider post message information about availability to message broker or event bus. Subscriber/consumer registers subscriptions with the brokers, letting brokers to perform the filtering. Brokers normally performs store and forward function to route messages in job service request/supply/ publish messaging with security offers from publishers to subscribers. In similar way subscriber in particular compartment/group send interim needs, required SR to the publisher through brokers. The process of rating, voting is taken care by broker based on rating priority [76-78].

- In Pub/sub systems data centers is registered with configuration and availability of space and resources/service during build time/initialization time or run time.
- Subscriber quote the SR, caste the vote based on the analysis provided by the broker and later deal with an agreement is reached based on consensus.
- When Pub publishes their availability, data triggered through the brokers to all subs who place the job/service request and select the required pub and negotiate their conflict if any.
- Resource discovery and matchmaking is done through brokers between pub/sub.
- Pub/sub's are allowed to remain in system topology and continue to operate normally regardless till the deal/agreement is reached in traditional way of client server paradigm.
- Pub/sub operate in distributed and parallel operations through message caching, tree based or network based routing and system scalable data centers with pub/sub infrastructure.
- With high load container use broker creates the channel for the resources to get allocated and load can be shared to other available PUB/SUB infra through distributed load sharing, which works on the basis of web syndication protocol.
- Load coupling is done by pub sub model. Pub/Sub should confirm once connected to track them using tracking table.
- Pub/sub is connected to a factory through loosely coupled system. This provides asynchronous communication for distributed systems in autonomous fashion.
- Subscribe based on context/topic, with mechanism of detection of overload/resources/service with enabled pub sub messaging with advance querying and queue based on expressed interest of service or resource accommodated with SR.
- Brokers negotiate polling voting for SR based on negotiation mechanism.
- Client register when overload occurs, it can be shared on the published availability of space to afford service.
- When SR arise put for rating, voting, polling and study of impact it is implemented and service offered helps in changing the physical location of VM periodically to counter attack as per subject of interest.
- Event framework can be used as event for need of security/load increase based on the event appeared new multilateral need access with predefined event publication.
- Pub/sub addresses is stored in catalog framework detects and publish such events in events driven manner as part of pub sub model.
- When overload occurred message is posted or published to a given queue and remote discovery framework setup and match frame occur and allocated in the similar way SR lined in queue and stored in factory then polled for voting and rating based on feasibility study security is offered.
- Rule engine is designed and the message is posted or published to a given queue an engine extracts the set of SR's defined in the queue on timely fashion with matching of space when over load occur.
- Tracking system changes, notification sent/posted to self identified systems, events need immediate actions.
- Fast and high throughput through flooding of information continuously using en-queue and de-queue.
- Persistent and desirable SR implementation with streamlining of Pub/Sub loosely coupled architectures.
- Real time producer consumer threads, easily available, immediately visible to consumer based on critical event base systems.
- Distribute consumption over a cluster of events (SR and Load demand) [76-78].

Like Honey bees single individual bees search of promising site, with occurring danger, based on collating independent decision encouraging minority views and requiring a democratic quorum to make final decision. Using the honey bees co-operation and effectual group decision without any bias course and scouts to pool information reflexively binds groups together into cohesive units [78]. Inspired by democratic view we proposed the MCF model for high secured needs of each individual [76-78] to be implemented.

4. MCF computing utility major characteristics

Cooperative computing is more accurate attribute of co-operative and collaborative mode of sharing computing resources and services over Internet using global MCF. MCF computing utility is the availability of a variety of clustered - computing servers, application servers, web servers and storage servers at unique place. Engineers and tech Savvies realized the need for homogeny of VM's and data centers. The need to inter-link computing grids, over geographically distributed high speed, secure digital communication networks and stratagem to optimize the distribution of computing power. The major characteristics of utility MCF are cooperative service vendor who owns and maintains infrastructure through multilateral relationship across distributed data centers accessed over browsers. MCF provides greater transparency, with traceability of men, money and material with several fold integrated into the business system. It helps to follow globally unique laws, regulations, and security measures, resolves jurisdictional issues through multilateral dialogue, operation, collaboration and co-operation. This cooperative

utility computing integrates multiple clusters of suppliers and providers, customers across multiple distribution channels with multiple fulfillments among the partners. It is boomed with forecast accuracy, return on investment. As the locations/jurisdictions of the data center in international boundaries, across the globe use MCF, this provides clarity in what laws apply in the case of disputes.

From the literature survey we conclude the security and privacy need to be considered at different stage of implementing MCF the same is depicted in the fig 4, is self explanatory. MCF provides availability, security, operability, metering, provisioning and deployment automation, utility auditing, cloud carrier, commissioning, orchestration inter-operability, resource abstraction, utility brokering. In addition, it also provides network-resource virtualization, VM management, projects and quotas allocation, dynamic provisioning, and integrated authentication. It also provides flexible networks, multi-container support, fast resource acquisition, geo-anonymity, replication, automated failure recovery. Also with self manageability, trustworthiness challenge, isolation management, agility, flexibility, transparency, best of breed, better reach, efficiency enhancement, accountability focus etc. Customer loyalty, collaborative and innovative environment, accelerating full potential for deployment, and follow unique standards. It also provides, think-tank sharing and unique adoption, with pilot implementation of APPS, PROTOS, reference architectures, road maps etc, steering strategic directions, evaluate, execute the plans and computing services based on mutual business needs. It also helps in set up akin and faith; administer contracts and agreement, pricing, meeting the proof of concepts. In other words, aspire to make such ideas a reality; it helps with less access time of computer resources compared with the speed of light.

5. Related work

In literature we do not find much substantial work done in this regard, based on available literature/references we compare and contrast our work and try to improve the lacks using MCF. Ref [31] proposes no hype architecture, with removal of the hyper-visor (VMM) from the virtualization layer. Still research needs to be done in depth in this direction. The work in [53] describes historical perspective, technical details, and main implementation methods for X86 architectures independent of vendors. But [53] work lacks to throw the light on security and privacy related architectural details. Using MCF we try to address these issues, and provide an end to end secured communication between the modules. In [6] authors explains the VM security breaches, threats, issues and provides generalized recommendations to achieve secured virtual network architecture. The work in [56] presents placing diverse consumers, workloads on the identical virtualization platform under multitenant. [56] Lacks proper architectural details. No comparisons to prove the novelty of work, it's based on illusions and empirical work, no light on interface, and time computational efficiency. The work on federated computing systems states the challenges, with the need to be consider in the future Internet to be achieved [78]. This serves the basis to realize MCF vision [78]. a) Security and Trust: Partners and stakeholders to access resources to set trust and agreements without any form a centralized authority with the need of decentralized and distributed system security and trust mechanisms. Establish stabilized trust management systems. b) Syntactic and semantic interoperability and representation of the models with context of information across interconnected network domains, connected dynamically and automatically managed, aligned to federation of data centers. Mapping, with identification, allocation and provisioning techniques across the geography of networks. c) Agreement negotiation and alignment: Participating parties need to come to an agreements with associated costs, benefits, goals and federation wide policies by avoiding conflicts, achieve policy view goals, aligned by negotiations, protocols, with compromise and agreement is finalized. d) Resource discovery and match making, configuration of specific capabilities, description, resource configuration, postulate the need for match making and discovery mechanisms. The abstract capabilities and configurations must be mapped into actual physical or virtual resources to offer the required functionality. This is achieved through scalable and distributed mechanisms that allow shared capabilities and resources satisfy specific requirements to discovered and configured resources and track them through log table using intelligent transformation mechanisms and mapped goals. e) Management coordination: Co-operation among varied data centers, stakeholders to achieve secured service benefits. Ref [78] proposes the empirical architecture without any proof or justification.

5.1 Merits of using MCF

We overcome the lacks in [56] using MCF as follows: a) Abstract: procure the varied security requirements from consumers, b) Consolidate: analyze for the feasibility, c) identify the conflicts, threats leading to erratic and impulsive effects, d) resolve and negotiate agreed and conferred levels of security goals among the input security requirements by many customers and enforce it, e) accommodate the requirements of the varied clustered customers, f) pool legal moves available at any stage of negotiation process. When the consumers start, create or run the VM's express security preferences, one as to develop the MCF through negotiation using voting and rating, g) clustering by grouping the requirements from the end users with common goals and choose with maximum need of security requirements from the individual/group of end users, h) collaborating- pooling VM's security needs through, communication, voting, rating, requirements. We use MCF by defining, accreditations, and standards, high performance, and high availability of standard hybrid security strategies, suiting VM's. To provide better feasible solution implemented using multilateral collaboration, co-operation, communication, coordination, among the users

and group, through consensus integration of needs using brokers (pub-sub model) in unified framework. Co-operative computing removes monopoly and duopoly benefits masking through services. The overall activities are shown in the below framework in fig 7 is self explanatory.

5.2 Outcome

In order to realize the implementation of MCF, we need dynamic discovery of VM, and using dynamic VM migration, on time clustering and offline clustering is done through multilaterally collaborated, coordinated, co-operatively clustered VM Management. The workload management done through negotiation for security requirement based on MCF using (pub sub model). We realize the properties of MCF by abstraction and virtualization with incredible complexity. This computing system exists and continues to evolve with good interfaces, interoperability. This utility allows the authorized customer to use the computers, applications, with consultancy storage infrastructure at anytime, anywhere across the world. This utility also helps to follow unique policies, standards, rules and regulations on use, charging policy quoted to customers for the use of the physical infrastructure facilities and services. The overall summary of secured MCF is presented in below fig 8. To realize MCF it requires to setup unique monitoring agency to frame standards, plans, policies and procedures and laws to operative under the proposed MCF. We can provide the unique security frame work using 9 d 's - deny, disrupt, defend, destroy, degrade, decrease, decide, decoy, defoliate using MCF. This MCF is based on law of democracy- representing the computing system - for the people, by the people, to the people and of the people [81]. Also using plan-do-act based on multilateral co-operation; we can narrow down or zero down the security threats. We have developed the prototype of our collaboration and cooperation based security management using MCF, we evaluated the system performance using the computing system having multiple tenancy/stakeholders.

5.3 Demerits:

Political and Economic issues require high standards, of physical robustness, high quality and military measures with more initial investment. Multiple nations should take collective global steps – no single nation may remedy. Globally each stakeholder should implement no harm policy and collective steps imposed domestically and internationally on public, private infrastructures. Services used by multiple vendors provide physical security. Nations will share the identity and security information exchange in structure exchange and provide the facility to network service provider. Nations should impose obligation for need and to innovate, implement new protocols. Need to know the techniques and operating practices that promote interoperability. In addition, with in limited network connectivity, over clarification of information, difficulty in applying the SR of all stakeholders. Establishing the trustworthiness is important and difficult. To negotiate one must determine a set of candidate federation partners based on their offered capabilities, domains need to know prior, with large network domains of large as Internet consisting of thousands of independently managed network domains with million soft and hardware resources, this may be an infeasible solution, if dedicatedly implemented can be fruitful solution.

6 Conclusions

The proposed multilateral co-operative computing utility framework helps in realizing the benefits of grid, cloud, distributed, HPC and parallel computing is clone of its entire predecessor computing technologies. By unique joint management and operation of the combined shared resources, apps, and services under MCF. Multilateral joint cooperative using unique complex security measures and guard against security threats, defects, fraudsters, hackers, cyber attacks, vulnerabilities and leakages. Better utilization of computing resources to maximum extent; Provides better reliability and disaster recovery backup's continuously; helps in reduction of human resources, space for datacenter required. Promotes green ecological computing, with carbon free emission footprint computing, quite - silent computing, solar power computing, and low power computing, lead free computing, cut down in cooling resources, usage of energy efficient systems. Promotes use of generic software's; Provides global computing technology, promotes usage of biodegradable, renewable source of energy for computing, and recycle of computer components, reduction of disposal; Limits usage of hazardous substance in semiconductors. We have presented our layered MCF architecture, where the security of each layer needs to be concentrated. SR belonging to geographically distributed stakeholder is implemented and executed using MCF unique architecture. This shared responsibility helps in promoting solidarity among competitors and fight against the malicious users. This work helps to capture the security requirement of each user using voting and rating method, based on cooperation and collaboration among each stake holders (multiple tenants) to have knowledge of internal and external source of the security breaches and platform used in public computing utility under unique framework applicable to all areas. MCF pave the way to research on security engineering and security control development process to develop complex flexible service to fit the need of each utility service consumer. And concentrate on automated security control and configuration engineering.

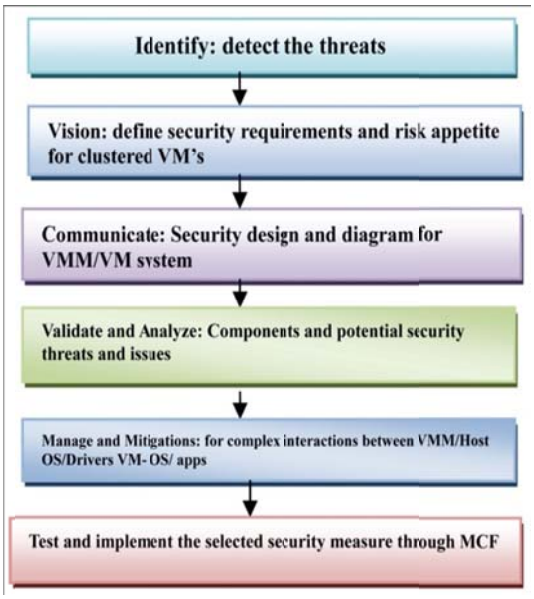


Fig 6 flow chart showing the steps to implement MCF based on [6]

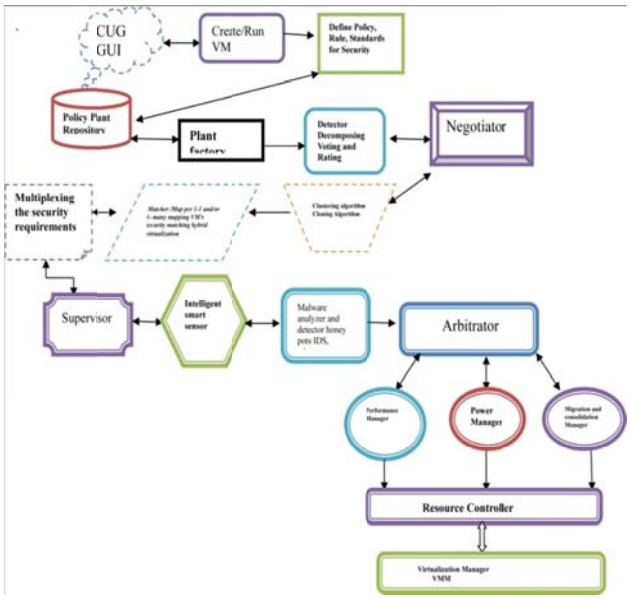


Fig 7 showing over all activities of MCF

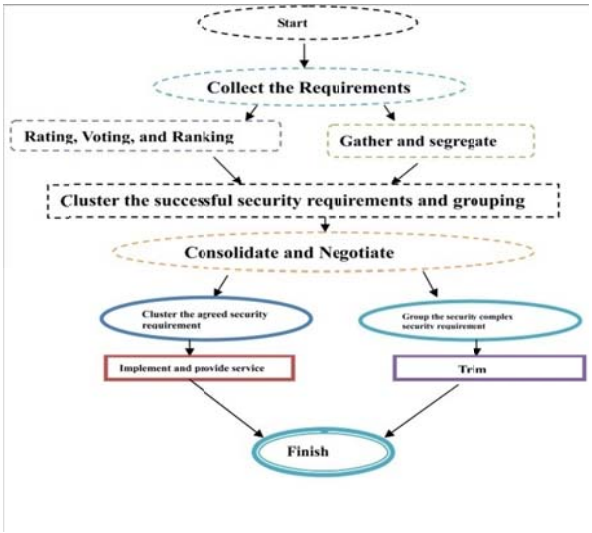


Fig 8 shows the overall steps involved in MCF

REFERENCES

[1] V. Rajaraman, "Cloud Computing", (p.242), General Article, Volume 19 - Issue 3, Resonance, journal of science education, Indian Academy of science education, March 2014.

[2] V Rajaraman," Utility Computing and Cloud Computing", Inaugural address, 2nd International Conference in Advances in Cloud Computing, Computer Society of India, Bangalore, Sept.19, 2013.

[3] V.A.Vyssotsky, F.J.Corbato, R.M.Graham, "*MULTICs operating system"*, Full Joint Computer Conference, Proc. AFIPS conference 1965 (Available at www. multics.org/fjcc3.1).

[4] Popek, G. J. AND Goldberg, "Formal requirements for virtualizable third generation architectures", Communications, ACM 17, 7, 412–421, R. P 1974.

[5] CERN LHC Computing Grid Project: http://lhcgrid.web.cern.ch/LHC grid/, accessed on Sept 9th 2014.

[6] Pearce, M., Zeadally, S., and Hunt, R. "Virtualization: Issues, security threats, and solutions", ACM Comput. Sur, Vol 45, No 2, 39 pages, Article 17, February 2013.

[7] Michael Armbrust , Armando Fox , Rean Griffith , Anthony D. Joseph , Randy H. Katz , Andrew Konwinski , Gunho Lee, David A. Patterson , Ariel Rabkin, Ion Stoica, Matei Zaharia, "Above the Clouds: A Berkeley View of Cloud Computing", Technical Report No. UCB/EECS-2009-28, http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html , February 10, 2009.

[8] Pradeep Padala, Kai-Yuan Hou, Kang G. Shin, Xiaoyun Zhu, Mustafa Uysal, Zhikui Wang, Sharad Singhal, Arif Merchant, "Automated Control of Multiple Virtualized Resources", Full text- Approved for External Publication, Hewlett-Packard Development Company, L.P. External Posting Date: November 21, 2008.

[9] Ajay Gulati, Chethan Kumar, Irfan Ahmad, Karan Kumar, " BASIL: Automated IO Load Balancing Across Storage Devices", ACM-FAST'10, Proceedings of the 8th USENIX conference on File and storage technologies, Pages 13-13, USENIX Association Berkeley, CA, USA, 2010.

[10] Evangelos Kotsovinos, "Virtualization: Blessing or Curse? Managing virtualization at a large scale is fraught with hidden challenges", vol. 5 4, n o. 1, communications of the ACM, January 2011.

[11] Timothy Wood, Prashant Shenoy , K.K. Ramakrishnan , Jacobus Van der Merwe, " CloudNet: Dynamic Pooling of Cloud Resources by Live WAN Migration of Virtual Machines", University of Massachusetts, Technical Report 2010.

[12] Justin Moore, Jeff Chase, Parthasarathy Ranganathan, Ratnesh Sharma, "Making Scheduling "Cool": Temperature-Aware Workload Placement in Data Centers", HP Labs, and the U.S. National Science Foundation, USENIX Annual Technical Conference, 2005.

[13] Albert Greenberg, James Hamilton, David A. Maltz, Parveen Patel, "The Cost of a Cloud: Research Problems in Data Center Networks", editorial note submitted to CCR, ACM SIGCOMM Computer Communication Review archive Volume 39 Issue 1, Pages 68-73, January 2009.

[14] Sunay Tripathi, Nicolas Droux, Thirumalai Srinivasan, Kais Belgaied," Crossbow: From Hardware Virtualized NICs to Virtualized Networks", 2009 Sun Microsystems, Inc. VISA'09, Barcelona, Spain. ACM, August 17, 2009.

[15] Sunay Tripathi, Nicolas Droux, Kais Belgaied, Shrikrishna Khare, "Crossbow Virtual Wire: Network in a Box", www.usenix.org/events/lisa09/tech/full_papers/tripathi.pdf., 2009

[16] Piyush Shivam, Varun Marupadi, Jeff Chase,Thileepan Subramaniam, Shivnath Babu," Cutting Corners: Workbench Automation for Server Benchmarking", https://www.usenix.org/event/usenix08/tech/full_papers/.../shivam.pdf, 2008

[17] Krishna Kant, "Data center evolution A tutorial on state of the art, issues, and challenges", Published by Elsevier B.V., Computer Networks 53, 2009

[18] Austin T. Clements, Irfan Ahmad, Murali Vilayannur, Jinyuan Li," Decentralized De-duplication in SAN Cluster File Systems",USENIX'09 Proceedings of the 2009 conference on USENIX Annual technical conference, Pages 8-8, ACM. USENIX Association Berkeley, CA, USA, 2009.

[19] Harold C. Lim Shivnath Babu Jeffrey S. Chase, "Automated Control for Elastic Storage", ICAC'10, Washington, DC, USA, June 7–11, 2010.

[20] Fabien Hermenier Xavier Lorca Jean-Marc Menaud, Gilles Muller, Julia Lawall," Entropy: a Consolidation Manager for Clusters", 2009 ACM 978-1-60558-375-4/09/03.VEE'09, Washington, DC, USA, March 11–13, 2009.

[21] Anton Burtsev, Kiran Srinivasan, Prashanth Radhakrishnan, Lakshmi N. Bairavasundaram, Kaladhar Voruganti, Garth R. Goodson, "Fido: Fast Inter-Virtual-Machine Communication for Enterprise Appliances", , conference on USENIX, pages 25-29, https://www.usenix.org/legacy/event/usenix09/tech/slides/burtsev.pdf, USENIX '09 proceedings of 2009

[22] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung, Google." The Google File System", SOSP'03, , Bolton Landing, New York, USA, 2003 ACM, October 19–22, 2003.

[23] Senthilkumaran R and Purushottam Kulkarni, "Insight: A Framework for Application Diagnosis using Virtual Machine Record and Replay", www.cse.iitb.ac.in/internal/techreports/reports/TR-CSE-2014-57.pdf., technical report 2014

[24] Konrad Miller, Fabian Franz, Thorsten Groeninger, Marc Rittinghaus, Marius Hillenbrand, Frank Bellosa, "KSM++: Using I/O-based hints to make memory-de-duplication scanners more efficient", RESoLVE'12, London, March 3rd, 2012.

[25] Avi Kivity, Yaniv Kamay, Dor Laor, Uri Lublin, Anthony Liguori, "kvm: the Linux Virtual Machine Monitor", Proceedings of the Linux Symposium, Ottawa, Ontario, Canada, www.linux-kvm.com/sites/default/files/kivity-Reprint.pdf. June 27th–30th, 2007.

[26] Tathagata Das, Pradeep Padala, Venkata N. Padmanabhan, Ramachandran Ramjee Kang G. Shin, "LiteGreen: Saving Energy in Networked Desktops Using Virtualization", Microsoft research, research.microsoft.com/apps/pubs/default.aspx?id=131576, 23 june 2010.

[27] Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, , Christian Limpach, Ian Pratt, Andrew Warfield, "Live Migration of Virtual Machines", Volume 2, Pages 273-286, USENIX Association Berkley, CA, USA, 2005

[28] Jeffrey Dean and Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", research.google.com/archive/mapreduce-osdi04.pdf, 2004.

[29] Carl A. Waldspurger, "Memory Resource Management in VMware ESX Server", In Proc. Fifth Symposium on Operating Systems Design and Implementation (OSDI '02), Dec. 2002.

[30] Eric Keller, Jakub Szefer, Jennifer Rexford, Ruby B. Lee," NoHype: Virtualized Cloud Infrastructure without the Virtualization", ISCA'10, ACM, Saint-Malo, France, June 19–23, 2010.

[31] Samuel T. King, George W. Dunlap, Peter M. Chen, " Operating System Support for Virtual Machines", Proceedings of the 2003 USENIX Technical Conference, 2003.

[32] Ajay Gulati, Irfan Ahmad, Carl A. Waldspurger, "PARDA: Proportional Allocation of Resources for Distributed Storage Access", USENIX Association, 7th USENIX Conference on File and Storage Technologies, Jan 14, 2009.

[33] Akshat Verma, Puneet Ahuja, Anindya Neogi, "pMapper: Power and Migration Cost Aware Application Placement in Virtualized Systems", Middleware 2008, LNCS 5346, pp. 243–264, IFIP International Federation for Information Processing 2008.

[34] Michael R. Hines, Kartik Gopalan, "Post-Copy Based Live Virtual Machine Migration Using Adaptive Pre-Paging and Dynamic Self-Ballooning", VEE'09, Washington, DC, USA. 2009 ACM, March 11–13, 2009.

[35] Timothy Wood, Ludmila Cherkasova, Kivanc Ozonat, Prashant Shenoy, "Profiling and Modeling Resource Usage of Virtualized Applications", Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware, Pages 366-387, Middleware 2008.

[36] George W. Dunlap, Samuel T. King, Sukru Cinar, Murtaza A. Basrai, Peter M. Chen, "ReVirt: Enabling Intrusion Analysis through Virtual-Machine Logging and Replay", Proceedings of the 2002 Symposium on Operating Systems Design and Implementation (OSDI), Volume 36 Issue SI, Pages 211-224, ACM New York, NY, USA, Winter 2002.

[37] Diego Ongaro, Alan L. Cox, Scott Rixner, "Scheduling I/O in Virtual Machine Monitors", , Proceedings of the fourth ACM SIGPLAN/SIGOPS, VEE 2008.

[38] H. Andrés Lagar-Cavilla, Joseph A. Whitney, Adin Scannell, Philip Patchin, Stephen M. Rumble, Eyal de Lara, Michael Brudno, M. Satyanarayanan, "SnowFlock: Rapid Virtual Machine Cloning for Cloud Computing", EuroSys'09, Nuremberg, Germany ACM, April 1–3, 2009

[39] Akshat Verma, Ricardo Koller, Luis Useche, Raju Rangaswami, "SRCMap: Energy Proportional Storage using Dynamic Consolidation", USENIX, ACM, 23 Feb, 2010.

[40] Yaozu Dong, Xiaowei Yang, Xiaoyong Li, Jianhui Li, Kun Tian, Haibing Guan, "High Performance Network virtualization with SR-IOV", IEEE 16th International Symposium on HPCA, Jan 2010.

[41] Ben Pfaff, Tal Garfinkel, Mendel Rosenblum, "Virtualization Aware File systems : getting beyond the limitations of virtual disks", USENIX, ACM, 2006.

[42] Prajakta patil, Purushottam Kulkarni, Umesh Bellur, "Virtperf: A Performance Profiling Tool for Virtualized Environments", IEEE 4th, Internationala Conference on Cloud Computing, 2011.

[43] Timothy Broomhead, Laurence Cremean, Julien Ridoux, Darryl Veitch, "Virtualize Everything but Time", Usenix, 2010.

[44] Ripal Nathuji, Karsten Schwan, "VirtualPower : Coordinated Power management in virtualized enterprise systems", ACM, SOSP Oct 2007.

[45] James E Smith, Ravi Nair, "The Architecture of Virtual Machines", IEEE, 2005.

[46] Jian Wang, Kwame-Lante Wright, Kartik Gopalan, "XenLoop: a transparent high performance inter-VM network Loopback", Cluster Comput 12: 141–152, Springer, 2009.

[47] Sriram Govindan, Arjun R. Nath, Amitayu Das, Bhuvan Urgaonkar, Anand Sivasubramaniam, "Xen and Co.: Communication-aware CPU Scheduling for Consolidated Xen-based Hosting Platforms", ACM, VEE'07, San Diego, California, USA, June 13–15, 2007.

[48] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer,, Ian Pratt, Andrew Warfield, "Xen and the Art of Virtualization", ACM, SOSP'03, Bolton Landing, New York, USA, October 19–22, 2003.

[49] Keith Adams, Ole Agesen, "A Comparison of Software and Hardware Techniques for x86 Virtualization", ACM, ASPLOS'06 October 21–25, , San Jose, California, USA, 2006 .

[50] Akshat Verma, Gargi Dasgupta, Tapan Kumar Nayak, Pradipta De, Ravi Kothari, "Server Workload Analysis for Power Minimization using Consolidation", ACM, USENIX, Jun 14, 2009.

[51] Justin Moore and Jeffrey S. Chase, Parthasarathy Ranganathan, "Weatherman: Automated, Online, and Predictive Thermal Mapping and Management for Data Centers ", IEEE, 2006.

[52] Edouard bugnion, scott devine, Mendel rosenblum, Jeremy sugerman, Edward y. wang, "Bringing Virtualization to the x86 Architecture with the Original VMware Workstation", ACM Transactions on Computer Systems, Vol. 30, No. 4, Article 12, Publication date: November 2012.

[53] Mendel Rosenblum, Tal Garfinkel, "Virtual Machine Monitors: Current Technology and Future Trends", Published by the IEEE Computer Society, May 2005.

[54] Ben Pfaff, Tal Garfinkel, Mendel Rosenblum, "Virtualization Aware File Systems: Getting Beyond the Limitations of Virtual Disks", ACM Usenix, May 2006.

[55] Pengfei Sun, Qingni Shen, Liang Gu, Yangwei Li, Sihan Qing, Zhong Chen, "Multilateral security architecture for virtualization platform in multi-tenancy cloud environment", IEEE, 2013.

[56] V.A.Vyssotsky, F.J.Corbato, R.M.Graham, *Multics operating sytem*, Full Joint Computer Conference, Proc. AFIPS conference 1965 (Available at www. multics.org/fjcc3.1), 1965.

[57] F.Berman, G.Fox, A.Hey (Editors), *Grid Computing: Making the Global Infrastructure a Reality*, Wiley Press, N.Y, U.S.A, 2013.

[58] I.Foster, C. Kesselman and S.Tueke *The Anatomy of the Grid: Enabling Sealable Virtual Organizations*, International Journal of High Performance Computer Applications, 15(3) : 200-222, 2001.

[59] I.Foster, C. Kesselman, *The Grid 2: Blueprint for a New Computing Infrastructure*, Morgan Kaufman, San Francisco, CA, USA, 2004.

[60] I.Foster, C. Kesselman, J. Nick and S.Tueke, *Grid Services for Disttributed system Integration*, IEEE Computer, 35(6): 37-46,2002.

[61] S.Tuecke et.al, Open Grid Services Infrastructure Version 1.0, Global Grid Forum Draft Recommendation, (See www. ggf.org), June 2003.

[62] Ian Gortona, Sanjeev Motwanib, "Issues in co-operative software engineering using globally distributed teams", Information and Software Technology 38 (1996) 647-655, Elsevier Science B.V, Received I6 December 1994; revised 19 December 1995, accepted 15 January 1996.

[63] T.Wittig, N.R.Jennings and E.H. Mamdani, "Archon: framework for intelligent co-operation", Intelligent systems Engineering, Autumn 1994.

[64] Linden Brown and Hugh Pattinson "Information technology and telecommunications: impacts on strategic alliance formation and management", Information Technology and Telecommunications, Management Decision, Vol. 33 No. 4, pp. 41-51 © MCB University Press Limited, 1995.

[65] Fuqing Yang, Hong Mei, "Development of Software Engineering: Co-operative efforts from academia, government and industry", ICSE'06, May 20-28, 2006, Shanghai, ACM ,China, 2006.

[66] Mark Walkley, Jason Wood, and Ken Brodlie, "A Distributed Co-operative Problem Solving Environment", http://www.comp.leeds.ac.uk, 2002.

[67] B. G. Evans and K. Baughan, "Visions of 4G", ELECTRONICS & COMMUNICATION ENGINEERING JOURNAL DECEMBER 2000, IEE: 2000 First received 2nd October and in final form 24th October 2000.

[68] James Broberg, Srikumar Venugopal, Rajkumar Buyya, " Market-oriented Grids and Utility Computing: The state-of-the-art and future directions", http://www.cloudbus.org/papers/MarketGridUtilityComp2001_2007.pdf. 2007.

[69] Manu A R, V K Agrawal, K N Bala Subramanya Murthy, "A study of security and privacy Issues of Cloud Ecosystem", International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868, Foundation of Computer Science FCS, New York, USA, Volume 5 – No. 5, April 2013.

[70] MR. Robert f. lentz, statement record version, u.s. house of representatives oversight and government reform committee subcommittee on management, organization and procurement, may 5, 2009.

[71] Kai Rannenberg, "Multilateral Security A Concept and Examples for Balanced Security", Pp. 151-162 in: Proceedings of the 9th ACM New Security Paradigms Workshop 2000, Cork, Ireland; ACM Press, September 19-21, 2000.

[72] J. Famaey and F. De Turck, "Federated Management of the Future Internet: Status and Challenges", international journal of network management, Int. J. Network Mgmt 2011; 00:1–21, Published online in Wiley Inter Science, 2012.

[73] Pub- Sub, , http://docs.oracle.com/cd/B10501_01/appdev.920/a96590/adg15pub.htm, accessed on Sep 9 2014

[74] Lothar Fritsch, Rani Husseiki, Ammar Alkassar,"An Analysis of Security Techniques and Technologies Complementary to Trusted Computing", Sirrix AG security technologies, A study on behalf of the German Federal Office for Information Security (BSI), Version 1.0 / 27. January 2010.

[75] Ross J. Anderson, "Security Engineering: A Guide to Building Dependable Distributed Systems", 2nd Edition,. http://www.cl.cam.ac.uk/~rja14/musicfiles/manuscripts/SEv1.pdf. March 2008

[76] Tania Banerjee Mishra, Sartaj Sahni, "PUBSUB: An Efficient Publish/Subscribe System", US Air Force Research Laboratory, under grant FA8750-11-1-0245, Copyright 20XX ACM.

[77] Ying Liu, Beth Plale, "Survey of Publish Subscribe Event Systems", 2003.

[78] Thomas D. Seeley, "Honey bee's democracy", Princeton university press, 2010.

[79] Manu A. R, Dr. V. K Agrawal, Dr. K N Bala Subramanya Murthy, "Representation of cloud ecosystem using engineering methodologies", international journal of computers & technology, VOL 7, NO 1, IJCT, 518-532, May, 2013:.

[80] Manu A R, Manoj Kumar M, Dinesha H A "An Approach to Enhance Security of Cloud Computing Services using Software Engineering Model", IJCA April 2013 Edition, April 18, 2013.

[81] The Constitution of India, Ministry of Law and Justice, Govt of India., http://lawmin.nic.in/coi/coiason29july08.pdf, As modified up to the 1st December, 2007

[82] Manu A. R, V. K Agrawal, K N Bala Subramanya Murthy, "A User Identification Technique to access Big Data using Cloud Services", Volume 91 - Number 1, IJCA, 2014 , Presented in Doctoral Symposium, DEBS-ACM Symposium- 2014, http://www.cse.iitb.ac.in/debs2014/wp-content/uploads/2014/04/DEBS-Schedule.pdf, ACM sigsoft, 8th international Conference, DEBS, IITB, Mumbai, India,May,26 – 29, 2014.

Author Index

A
Abraham, J. 74
Aggarwal, A. 155
Agrawal, V. K. 161
Alexakis, S. 1
Alrokayan, M. 49
Awasthi, C. 93

B
Bar, H. 1
Bauer, M. 1
Bawa, S. 122
Bhanu S., M. S. 17
Binu, V. 57
Bondada, M. B. 17
Buyya, R. 49, 141

C
Challa, N. R. 110

D
D., H. 65
Dara, S. 134
Dastjerdi, A. V. 49
Dietrich, P. 87
Domanal, S. G. 70
Dsouza, K. J. 117

G
Gangadhar, N. D. 57
Ganguli, M. 25
Goyal, S. 122
Guddeti, R. M. R. 70
Gupta, P. 25

J
Juan-Verdejo, A. 1
Justino, T. 141

K
Kafeza, E. 41
Kafeza, I. 41
Kanungo, P. 93
Kaur, D. 155
Khanna, R. 25
Kishore, N. 99
Kushwaha, V. 33

L
Lakshmanan, N. 65

M
M., M. K. 161
M., S. 65
M., S. 117
Mane, P. 106
Mehta, S. 74
Millham, R. 82
Mittal, D. 155
More, D. 74
Murthy, K. N. B. 161
Murugan, A. 87
Murugan, G. 9
Mylaraswamy, D. 87

N
Nagojappa, N. S. 99
Narayan, A. S. 25
Nuthula, V. 110

P
P., J. C. 149
Panas, E. 41
Patel, V. 65
Pathak, P. 74

R
R., A. 25
R., M. A. 161
Ratnaparkhi, A. 106

S
Sahni, S. 128
Sarvabhatla, M. 9
Shashidhara, M. S. 149
Simmhan, Y. 33
Singh, B. 122
Surajbali, B. 1

T
T. S., A. 70

V
Varma, V. 128
Vorugunti, C. S. 9

W
Walase, L. 74

X
Xu, B. 87

Y
Yeshodara, N. S. 99