TOWARDS DEADLINE GUARANTEED CLOUD STORAGE SERVICES

Preethi.s ,Raja.A

IV Sem MTech , Assistant Professor in Department of Computer Science and and Engineering
CByre Gowda Institute ofTechnology
Kolar,India
[1] Department of Computer Science and and Engineering
CByre Gowda Institute ofTechnology
Kolar,India

Abstract: Many organizations transfer their data and workload to commercial cloud storage systems. However, present unpredictable data access latency to tenants the multiplexing and sharing of the resources in a cloud storage system , which may make online data-intensive applications unable to satisfy their deadline requirements. Thus, it is important for cloud storage systems to provide deadline guaranteed services. In this paper, to meet a current form of service level objective (SLO) that constrains the percentage of each tenant's data access requests failing to meet its required deadline below a given threshold, we build a mathematical model to derive the upper bound of acceptable request arrival rate on each server. We then propose Deadline Guaranteed storage service (called DGCloud) that incorporates three basic algorithms. Its deadline-aware load balancing scheme redirects requests and creates replicas to release the excess load of each server beyond the derived upper bound. Its workload consolidation algorithm tries to maximally reduce servers while still satisfying the SLO to maximize the resource utilization. Its data placement optimization algorithm re-schedules the data placement to minimize the transmission cost of data replication. We further propose three enhancement methods to further improve the performance of DGCloud. A dynamic load balancing method allows an overloaded server to quickly offload its excess workload. A data request queue improvement method sets different priorities to the data responses in a server's queue so that more requests can satisfy the SLO requirement. A wakeup server selection method selects a sleeping server that stores more popular data to wake up, which allows it to handle more data requests. Our trace-driven experiments in simulation and DriveHQ show the superior performance of DGCloud compared with previous methods in terms of deadline guarantees and system resource utilization, and the effectiveness of its individual algorithms.

IndexTerms **– Cloud Storage, SLO, Deadline, Resource Utilization.**

I. INTRODUCTION

Distributed computing is enormously affecting how associations deal with their data innovation assets. Existing Cloud registering arrangements have not been worked in light of interoperability [40]. They more often than not bolt clients into a solitary Cloud foundation, stage or administration averting the immovability of information or programming made by them. Besides, the fight for strength between the enormous merchants, similar to Amazon, Google and Sales Force make them hesitant to concede to generally acknowledged gauges advancing their own, contrary arrangements. Interoperability is the missing component that will cure this circumstance and advantage both Cloud clients and Cloud suppliers [35]. Specifically, in an interoperable Cloud condition clients will probably contrast and pick among Cloud contributions and various qualities while they will switch between Cloud suppliers at whatever point required without setting information and applications in danger. The plenitude of simple to access processing assets empowered by distributed computing gives critical chances to associations, however stances challenges for ventures in various zones [36].

The present distributed computing scene comprises of an assorted arrangement of items and administrations that range from framework administrations (IaaS) to explicit programming administrations (SaaS) to improvement and conveyance stages (PaaS), and some more. The assortment of cloud administrations has prompted restrictive designs and advances being utilized by merchants, expanding the danger of seller lock-in for clients [14]. Cloud administration clients need to stay away from the issue of lock-in, where they risk being attached to a specific cloud specialist co-op because of the trouble and expenses of changing to utilize proportional cloud administrations from different suppliers. For instance, consider an association utilizing a PaaS (Platform as a Service). A PaaS stage from a specific merchant could bolster just restricted and exclusive web systems, dialects, libraries, databases, and so forth [5]. This can lead associations to create application structures directed by highlights offered by the PaaS cloud specialist co-op which can prompt their applications being bolted to that seller, basically non-convenient. There are no ideal answers for totally stay away from these issues, yet associations need to consider this issue cautiously when choosing cloud administrations [7].As ventures move to receive distributed computing in it different indications, the issues of interoperability should be tended to head on by the two suppliers and clients. To relieve the danger of lock-in associations should audit existing information administration and buy approaches and procedures to check whether improving information accessibility and (or) information get to productivity [9–17] in the cloud don't give due date ensures. In this paper, we take burden adjusting to fulfill the content due date prerequisites from various occupants with limited vitality and dispatch cost for a business distributed storage administration. The capacity frameworks we aim have little size articles (e.g., 100kB, for example, key-esteem stockpiling [18]. In particular, we suggest a Deadline Guaranteed to stockpile administration i.e., compels the percentage of each occupant's information access solicitations neglecting to comply with its required time constraint beneath a given limit. This goal is non-inconsequential in light of the fact that the solicitation conveyance and reproduction portion among servers is mind-boggling, and information prominence, server limits and occupant due date prerequisites are heterogeneous. To deal with it, in view of the queueing hypothesis, we scientifically determine the upper bound of worthy solicitation entry amount on every server to fulfill the SLOs everything being equal. We propose three calculations in DGCloud: (1) A heap adjusting the calculation to guarantee that the solicitation landing speed on every server is no greater than its upper bound. This calculation consolidates information demand divert and new reproduction distribution progress load from over-burden servers to under loaded servers. (2) An outstanding task at hand union calculation to amplify the framework usage and vitality effectiveness. It

modifies the information arrangement calendar controlled by our heap adjusting calculation, which decides the solicitation diverting and information position (i.e., which servers stores an information reproduction) to limit the numeral of servers being used. (3) An information position streamlining calculation to limit the dispatch cost for information replication. It respects the information arrangement improvement issue as a base weight impeccable coordinating issue [20] by considering the new reproduction portion as the change. Distributed computing is an advantageous, on-request model for system enter to a common pool of layout recording assets that can be rapidly supplied and release with insignificant administration exertion or particular organization association. Adroitly it alludes to a design of adaptable, ongoing, web-oriented data transformation administrations and benefits, achieving the significant requirements of clients, in absences of the clients submitting the cost in charging basic framework

We three calculations in DGCloud: (1) a heap adjusting the calculation to guarantee that the solicitation entry price on every attendant is no greater than it's higher bound. This calculation joins information demand reversing and recent imitation portion to pass load from over-burden server's i.e. Date server to relocated server. (2) An outstanding burden solidification calculation to boost framework usage and vitality proficiency. It modifies the information arrangement timetable dictated by our heap adjusting calculation, which decides the solicitation diverting and information position to limit the amount of servers being used. (3) An information situation streamlining calculation to limit the dispatch cost for information duplicate. It respects the information position enhancement issue as a base weight immaculate coordinating issue [20] by considering the new copy assignment as the change of information situation timetable to locate the ideal arrangement. We further propose three upgrade strategies to further enhance the presentation of DGCloud. (1) A effective burden adjusting strategy to enable an encumber server to rapidly offload its overabundance outstanding task at hand. (2) An information solicitation line improvement technique to set various needs to the information reactions in a server's line with the goal for many demands to fulfill the SLO prerequisite. (3) A wake-up server choice technique to choose a dozing server that stores progressively prevalent information to awake, which enables to hold many information demands.

II. PROBLEM STATEMENT AND SYSTEM OVERVIEW

A. System Model and Assumptions

We consider a heterogeneous cloud storage system consisting of M data servers, which may have different service capability and storage capacity. We assume that there are N tenants sharing the system. A data item consists of a number of data partitions. Each server may host a certain number of data partitions. Each data partition may have multiple replicas across several data servers to enhance the access efficiency and data availability [32], and each replica can be stored in any server. We assume that each data partition has at least r (r > 1) replicas. We suppose that the system maintains the consistency among replicas as [7], which is orthogonal to this work. A data request from a tenant arrives at the front-end servers of the cloud storage system first, and the loadbalancer assigns the request to the servers which hold the

TABLE 1: Notations.

	Description		Description
Sn	data server n, sn ∈ M	**dtk**	deadline of tk's request
tk	tenant k, tk ∈ J	**R(tk)**	set of servers for tk's data
L	total transmission cost	**Ptk**	probability of tk's beyond dtk
TR	threshold of service	**TU**	threshold of satisfaction level
MS	set of active servers	**ci**	data partition i, ci ∈ C

Use replicas of the requested data partitions. The service latency of a request is the duration between the arrival time at the front-end servers and the time when the response is returned to the front-end servers, including data transmission and processing latency. For a data request involving multiple data partitions, its latency is the longest service latency of a partition among all the data partitions. Each tenant tk ($1 \leq k \leq N$) has a deadline requirement for requests, denoted by dtk , which means that tk requires service latency on its requests to be no larger than dtk . If there are multiple types of requests from tk that have different deadlines, tk can be treated as several different tenants with different deadline requirements..

B. Problem Statement

In this paper, we introduce a form of SLOs [19] with deadline guarantees for cloud storage services. That is, for any tenant tk, no more than tk percent of all data requests have service latency longer than a given deadline dtk . Such an SLO is denoted by (tk , dtk). For example, Amazon DynamoDB [32] should guarantee that no more than 0.1% of its requests have a response time exceeding 300ms [19]. In order to satisfy the SLOs of all tenants, our deadlineaware load balancing problem is how to dynamically create data replicas in servers and redirect the requests to replicas such that the service latency of any request from tenant tk satisfies (tk , dtk). For easy reference, we list the major notations used in the paper in Table 1.

C. System Overview

Our final deadline guaranteed cloud storage system (DGCloud) incorporates three algorithms: deadline-aware load balancing algorithm (Section 4), workload consolidation (Section 5.1), and data placement optimization (Section 5.2). DGCloud is employed in the load balancer in current commercial cloud storage systems. The load balancer needs to periodically estimate the

parameters in Section 4.1 and check if the deadline-aware load balancing algorithm should be activated. If yes, the load balancer runs this algorithm off-line to generate the data placement schedule. Then, if the system resource utilization is low, it runs the workload consolidation algorithm to reduce the number of active servers in order to increase the system resource utilization while still guaranteeing the required SLOs of tenants. After the algorithm execution, a new data placement schedule and new request rate on each data replica are generated. To minimize the transmission cost for data replication, the data placement optimization algorithm is conducted to find the optimal transformation. Finally, data is replicated based on the final data placement schedule. We introduce each algorithm of DGCloud below

III. DEADLINE-AWARE LOAD BALANCING

To satisfy the deadline (SLO) requirements of all tenants, we need to balance the workload among all servers to avoid overloaded servers. The load balancing is usually a knapsack problem which is NP-hard [33]. Load balancing in cloud storage services is challenging since different servers have different service capabilities, different workloads have different deadline requirements and different data has different popularity. In this paper, we provide a heuristic deadline-aware load balancing scheme. We define overloaded servers as the servers that cannot satisfy the SLOs of all tenants, and define underloaded servers as the servers that can accept more data requests under the SLO requirements of all tenants. The basic idea of our load balancing scheme is to shift requests from the most overloaded servers to the most underloaded servers. The workload shifting is achieved by redirecting requests and creating new data replicas in other servers. First, the load balancer attempts to redirect arriving requests originally targeting overloaded servers to underloaded servers. After the request redirection scheduling, if some servers still cannot satisfy tenants' SLOs, new replicas are created in other servers, which will handle part of the requests. To design such a scheme, a critical problem is how to quantify the service capability of servers with regard to the tenants' SLOs. To handle this problem, we introduce two concepts: deadline-guaranteed request arrival rate and available service capacity as defined below.

A. Deadline-Guaranteed Request Arrival Rate Derivation

Our load balancing scheme requires the following parameters: i) the request arrival rate at each server sn and at each data partition replica ci in sn, denoted by λsn and λ ci sn , respectively, for computing available service capacity, ii) sn's service rate iii) the deadline guaranteed request arrival rate

B. Scheme Description

The load balancing scheme first computes each server sn's deadline-guaranteed request arrival rate λ 0 sn and then available service capacity asn . The data servers that have positive asn values are stored into a list named allocatable server list in descending order of the asn . Giving higher priority to servers with higher asn in assigning overloaded servers' excess load helps quickly release their load. For the servers having the same asn , they are stored in ascending order of their available storage capacity. The servers in the list are fetched one by one in the top-down manner to receive excess workload from overloaded servers. The ascending order of available storage capacity enables us to assign an excess workload to the best-fit server that has the least remaining storage capacity after the workload assignment in general. If a workload is assigned to a server that has high storage capacity, the remaining storage capacity may be larger but not large enough to support another workload, thus generating storage fragmentation. Even if the remaining storage capacity is big enough to support another workload, due to hosting the previous workload, it may not have enough service capacity to support another excess workload. Therefore, through our ordering strategies, we can avoid storage fragmentation and the situation in which a server cannot utilize its available storage capacity due to lack of service capacity.

IV. PERFORMANCE EVALUATION

In simulation. In this section we measure the performance of DGCloud in our lab-made simulator, which is written in JAVA. We implemented the simulator on Ubuntu OS with our own computer (i7-6700K, 32GB RAM and 512 SSD). We implement a three-level fat tree topology with edge switch, aggregation switch and core router. Each edge switch connects 30 servers and each four aggregation switches connect 100 edge switches. We simulated each data server separately by an instance of the same data server class, and accordingly simulated the entire network. We also simulated a namenode, which stores the request ratio and data replica distribution among servers. It also runs the *algorithm periodically to generate the new request ratio distribution and new replica*

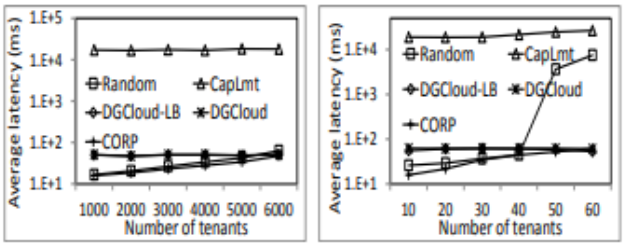

Fig:1 Average service latency vs. the number of tenants.

distribution schedule and sends out the schedules to all data servers. There were 30000 data servers in the cloud storage system. The storage capacity of each server was randomly chosen from {6TB, 12TB, 24TB} as [13, 41].

On Amazon EC2. We repeated the experiments in a real test environment consisting of 33 nodes in an availability zone of EC2's US west region [43]. We randomly chose 3 nodes as front-end servers on EC2, which generate the visits with the same rates as in [42]. The size of read/write has the same distribution as in [42]. The others are used as data servers with service rate randomly chosen from [6, 12], and particularly each node in EC2 simulates 10 data servers for enlarging scale. We use another server (which can be the distribute storage systems namenode when implementing in practice) to collect the request rate towards data partitions from all data servers. It runs the algorithm at beginning of each period, and sends out the data replication scheduling to all data

servers. The data servers will perform replication when they have spare networking resources. The centralized server also calculates request ratio distribution and data allocation schedule. Due to the storage limitation of VMs in the tested, the size of a partition and the storage capacity of a data server in our cloud storage system are reduced to 1/3000 of their previous settings to fit into the limited hard disk storage. The default number of tenants is 10. We measured the distance of any pair of data servers by the average ping latency.

V. CONCLUSIONS

In order to improve the deadline guaranteed performance in cloud storage services, in this paper, we first propose a deadline-aware load balancing scheme. It dynamically redirects requests and creates data replicas in servers to ensure a current form of SLO. We enhance our scheme with work consolidation to maximize the system resource utilization, and data placement optimization to minimize the transmission cost in data replication. We further propose three enhancement methods to further improve the performance of DGCloud. Our enhancement methods also reduce energy cost and transmission cost of data replication. In our future work, we will design a load balancing scheme that dynamically redirect requests and replicate data to ensure SLO under a request burst. Also, we will make DGCloud be suitable for other storage systems such as the Hadoop file system in the MapReduce platform.

REFERENCES

[1]. A. SHETH AND A. RANABAHU, "SEMANTIC MODELLING FOR CLOUD COMPUTING. PART I AND II," IEEE INTERNET COMPUTING MAGAZINE VOL 14, PP. PP 81-83, 2010.

[2]. J. McKendrick, "Does Platform as a Service have interoperability issues?," 24 May 2010. Available: http://www.zdnet.com/article/does-platform-as-a-service-have-interoperability-issues/.

[3]. D. Jamil and H. Zaki, "SECURITY ISSUES IN CLOUD COMPUTING AND COUNTERMEASURES," International Journal of Engineering Science and Technology vol 3 no 4, pp. pp 2672-2676, 2011.

[4]. T.G.PeterMell,"http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf," september 2011.

[5]. A. V. Dastjerdi, S. G. H. Tabatabaei, and R. Buyya, "An Effective Architecture for Automated Appliance Management System Applying Ontology-Based Cloud Discovery," in Proceedings of the 10th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2010), Melbourne, Australia, 2010.

[6]. C. Wang, k. Ren and Q. Wang, "Security Challenges for the Public Cloud," IEEE computer society Issue No.01 - January/February (2012 vol.16), pp. pp: 69-73, 2012. Ghosh et al., International Journal of Advanced Research in Computer Science and Software Engineering 5(2), February - 2015, pp. 910-913 © 2015, IJARCSSE All Rights Reserved Page | 913

[7]. Hannay, J.E., Sjøberg, D.I.K.: A Systematic Review of Theory Use in Software Engineering Experiments. Journal of IEEE Transaction on Software Engineering 33(2), 87–107 (2007)

[8]. Harauz, J., Kauifman, M., Potter, B.: Data Security in the world of cloud computing. IEEE Security & Privacy 7(4), 61–64 (2009)

[9]. Takabi, H., Joshi, J.B.D.: Security and Privacy Challenges in Cloud Computing Environments. Published. IEEE Security and Privacy 8(6), 24–31 (2010)

[10]. Bernstein, David; Ludvigson, Erik; Sankar, Krishna; Diamond, Steve; Morrow, Monique

[11]. B. Rochwerger et al., "*Reservoir—When One Cloud Is Not Enough*," Computer, Mar. 2011, pp. 44-51.

[12]. M.P. Papazoglou and W. van den Heuvel, "*Blueprinting the Cloud*," IEEE Internet Computing, Nov./Dec 2011, pp. 74-79.

[13]. Foued Jrad, Jie Tao and Achim Streit Steinbuch Centre for Computing, Karlsruhe Institute of Technology, Karlsruhe, Germany {foued.jrad, jie.tao, achim.streit}@kit.edu.

[14]. D. Bernstein and D. Vij, "*Intercloud Security Considerations*," Proc. 2nd Int'l Conf. Cloud Computing (CloudCom 10), IEEE Press, 2010, pp. 537-544.

[15]. Bernstein, D., Vij, D.: Using XMPP as a transport in Intercloud Protocols In Proceedings of CloudComp 2010, the 2nd International Conference on Cloud Computing (2010).

[16]. Bernstein, D.: The Intercloud: *Cloud Interoperability at Internet Scale*, In Proceedings of the 2009 NPC, pp. xiii (Keynote 2), Sixth IFIP International Conference on Network and Parallel Computing (2009).

[17]. M. A. Morsy, J. Grundy and Müller I. "An Analysis of the Cloud Computing Security Problem" In PROC APSEC 2010 Cloud Workshop. 2010.

[18]. S. Ramgovind, M. M. Eloff, E. Smith. "The Management of Security in Cloud Computing",In PROC 2010 IEEE International Conference on Cloud Computing 2010.

[19]. Bernstein, D., Ludvigson, E., Sankar, K., Diamond, S., and Morrow, M., Blueprint for the Intercloud - Protocols and Formats for Cloud Computing Interoperability.. In Proceedings of ICIW '09, the Fourth International Conference on Internet and Web Applications and Services, pp. 328-336 (2009).

[20]. Haji Binali, Chen Wu, and Vidyasagar Potdar, "Web Data Migration: Connecting Databases in the Cloud," 7th IEEE International Conference on Computer Science and Technologies (IEEE DEST 2013), 2013.

[21]. Wang, C., Q. Wang, K. Ren and W. Lou,"Privacypreserving public auditing for data storage security in cloud computing",IEEE Computer Society, ISSN:0018- 9340, Vol.62,Issue 2,page-362-375, 2010.

[22]. Venkatesh, M., M.R.Sumalatha and C.Selva Kumar,"Improving public auditability, data possession in data storage security for cloud computing."International Conference on Recent Trends in Information Technology (ICRTIT),IEEE,ISBN:978-1-4673-1599-9, page-463-469,2012. (IOSRJCE)

[23]. Rashmi Rao, Pawan Prakash, "Improving security for data migration in cloud computing using randomized encryption technique "IOSR Journal of Computer Engineering e-ISSN: 2278-0661, p- ISSN: 2278-8727,Volume 11, Issue 6, PP 39-42,2013 .

[24]. Prashant Pant, Sanjiv Thakur,"Data Migration Across the clouds", International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-3, Issue-2,
May 2013.
[25]. Virendra Singh Kushwah, Aradhana Saxena," A Security approach for Data Migration in Cloud", International Journal of Scientific and Research Publications, Volume 3, Issue 5,
ISSN 2250-3153,2013.
[26]. Wei Hao, I-Ling Yen and Bhavani Thuraisingham, ,"Dynamic Service and Data Migration in the Clouds", Computer Software and Applications Conference,COMPSAC,
33rd Annual IEEE International Conference, ISSN: 0730-3157, Page:134-139,2009.
[27]. K.Meenakshi, Victo Sudha George 2014 " Cloud Server Storage Security Using TPA" International Journal of Advanced Research in Computer Science & Technology
(IJARCST), ISSN : 2347 - 9817 (Print), Vol. ,2 Issue Special 1, Jan-March 2014.
[28]. Balakrishnan.S, Saranya.G, Shobana.S, Karthikeyan. S 2011 "Introducing Effective Third Party Auditing (TPA) for Data Storage Security in Cloud"IJCST,Vol.2, Issue-2,
ISSN:2229-4333,page-397-400,2011.
[29]. D.Srinivas " Privacy-Preserving Public Auditing In Cloud Storage Security",(IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (6),
page-2691-2693, 2011.
[30]. Pravin O. Balbudhe, "Cloud Storage Reference Model for Cloud Computing," International Journal of IT, Engineering and Applied Sciences Research, vol. 2, pp. 81–85,March 2013.
[31]. Foster I, Zhao Y, Raicu I, Lu S (2008) Cloud Computing and Grid Computing 360-Degree Compared. In: Grid Computing Environments Workshop (GCE'08).
[32]. S. Overby, How to Negotiate a Better Cloud Computing Contract, CIO, April 21, 2010, www.cio.com/article/591629/How_to_Negotiate_a_Better_Cloud_Computing_Contract.
[33]. A Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure Functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In
Advances in Cryptology - EUROCRYPT 2010. Springer, 2010.
[34]. An article on "Predictions about the future of Cloud Computing" available at http://mediastar91.blogspot.in/2012/04/predictions-about-future-ofcloud.html.
[34]. An article on "Predictions about the future of Cloud Computing" available at http://mediastar91.blogspot.in/2012/04/predictions-about-future-ofcloud.html.
[35]. M. Pirretti, P. Traynor, P. McDaniel, and B. Waters, "Secure attribute-based systems", Journal of Computer Security, vol. 18, no. 5, pp. 799–837, 2010.
[36]. Qingni Shen; Lizhe Zhang, Xin Yang, Yahui Yang, Zhonghai Wu, Ying Zhang, "SecDM: Securing Data Migration between Cloud Storage Systems," Dependable, Autonomic and
Secure Computing (DASC), 2011 IEEE Ninth International Conference on,
vol., no.,pp.636,641, 12-14 Dec. 2011.
[37]. Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia, "A
view of cloud computing," in 2010 Communications of the ACM,2010, vol-53,pp-50-58.
[38]. S Carlin, K Curran, "Cloud Computing Security," in 2009 International Journal of Ambient Computing and Intelligence.
[39]. Ali Khajeh-Hosseini, David Greenwood, Ian Sommerville,"Cloud Migration: A Case Study of Migrating an Enterprise IT System to IaaS" in 2010 IEEE 3rd International Conference
on Cloud Computing CLOUD, pp. 450-457.
[40]. Rakesh Sachdeva, Prabhpreet Kaur "SSM: Secure Storage Migration among Cloud Providers" in The International Journal Of Science & Technoledge. Vol 2, issue 3, March
2014, pp. 120-126.
[41]. Cloud Computing", MIPRO, 2012 Proceedings of the 35th International Convention, IEEE, 21-25 May,2012.PP 1484-1489.
[42]. P. Mell and T. Grance, NIST Working Definition of Cloud Computing, Version 15, 7 October 2009. Accessed on Jan 3, 2011:http://www.nist.gov/itl/cloud/upload/cloud-
def-v15.pdf .
[43]. C4ISR Architecture Working Group Interoperability Panel, Levels of Information Systems Interoperability (LISI), Department of Defense, Washington, DC, 30 March 1998.
[44]. J. Urquhart, "Exploring cloud interoperability, part 2", news.cnet.com, May 7, 2009. Accessed on March 8, 2010:http://news.cnet.com/8301-19413_3-10235492-240.html