p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 13
September 2016

Load balancing an application scaling for the cloud eco system

Ms. R.Samatha rani

Ms.B.Vishwaja

ABSTRACT:

In this paper we introduce an energy-aware operation model used for load balancing and application scaling on a cloud. The basic philosophy of our approach is defining an energy-optimal operation regime and attempting to maximize the number of servers operating in this regime. Idle and lightly-loaded servers are switched to one of the sleep states to save energy. The load balancing and scaling algorithms also exploit some of the most desirable features of server consolidation mechanisms discussed in the literature.

EXISTING SYSTEM:

- ❖ An important strategy for energy reduction is concentrating the load on a subset of servers and, whenever possible, switching the rest of them to a state with low energy consumption. This observation implies that the traditional concept of load balancing in a large-scale system could be reformulated as follows: distribute evenly the workload to the smallest set of servers operating at optimal or near-optimal energy levels, while observing the Service Level Agreement (SLA) between the CSP and a cloud user. An optimal energy level is one when the performance per Watt of power is maximized.
- ❖ In order to integrate business requirements and application level needs, in terms of Quality of Service (QoS), cloud service provisioning is regulated by Service Level Agreements (SLAs): contracts between clients and providers that express the price for a service, the QoS levels required during the service provisioning, and the penalties associated with the SLA violations. In such a context, performance evaluation plays a key

role allowing system managers to evaluate the effects of different resource management strategies on the data center functioning and to predict the corresponding costs/benefits.

DISADVANTAGES OF EXISTING SYSTEM:

- On-the-field experiments are mainly focused on the offered QoS, they are based on a black box approach that makes difficult to correlate obtained data to the internal resource management strategies implemented by the system provider.
- Simulation does not allow to conduct comprehensive analyses of the system performance due to the great number of parameters that have to be investigated.

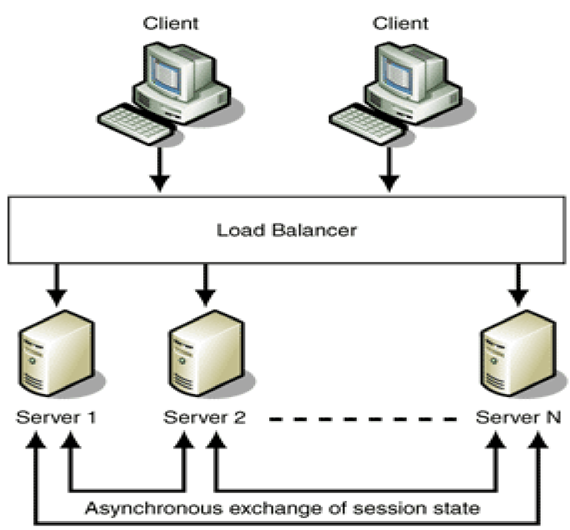
PROPOSED SYSTEM:

- ❖ There are three primary contributions of this paper:
- ❖ a new model of cloud servers that is based on different operating regimes with various degrees of \energy efficiency" (processing power versus energy consumption);
- ❖ a novel algorithm that performs load balancing and application scaling to maximize the number of servers operating in the energy-optimal regime; and
- ❖ analysis and comparison of techniques for load balancing and application scaling using three differently-sized clusters and two different average load profiles.
- ❖ The objective of the algorithms is to ensure that the largest possible number of active servers operate within the boundaries of their respective optimal operating regime. The actions implementing this policy are: (a) migrate VMs from a server operating in the undesirable-low regime and then switch the server to a sleep state; (b) switch an idle server to a sleep state and reactivate servers in a sleep state when the cluster load increases; (c) migrate the VMs from an overloaded server, a server operating in the undesirable-high regime with applications predicted to increase their demands for computing in the next reallocation cycles.

ADVANTAGES OF PROPOSED SYSTEM:

- After load balancing, the number of servers in the optimal regime increases from 0 to about 60% and a fair number of servers are switched to the sleep state.
- There is a balance between computational efficiency and SLA violations; the algorithm can be tuned to maximize computational efficiency or to minimize SLA violations according to the type of workload and the system management policies.

SYSTEM ARCHITECTURE:



1 INTRODUCTION

With the explosive growth of digital data, deduplication techniques are widely employed to backup data and minimize network and storage overhead by detecting and eliminating redundancy among data. Instead of keeping multiple data copies with the same content, deduplication eliminates redundant data by keeping only one physical copy and referring other redundant data to that copy. Deduplication has received much attention from both academia and industry because it can greatly improves storage utilization and save storage space, especially for the applications with high deduplication ratio such as archival storage systems. A number of deduplication systems have been proposed based on various deduplication strategies such as client-side or server-side deduplications, file-level or block-level deduplications. A brief review is given in Section 6. Especially, with the advent of cloud storage, data deduplication techniques become more attractive and critical for the management of ever-increasing volumes of data in cloud storage services which motivatesenterprises and organizations to outsource data storage to third-party cloud providers, as evidenced by many real-life case studies [1]. According to the analysis report of IDC, the volume of data in the world is expected to reach 40 trillion gigabytes in 2020 [2]. Today's commercial cloud storage services, such as Dropbox, Google Drive and Mozy, have been applying deduplication to save the network bandwidth and the storage cost with client-side deduplication. There are two types of deduplication in terms of the size: (i) *file-level deduplication*, which discovers redundancies between different files and removes these redundancies to reduce capacity demands, and (ii) *blocklevel deduplication*, which discovers and removes redundancies between data blocks. The file can be divided into smaller fixed-size or variable-size blocks. Using fixedsize blocks simplifies the computations of block boundaries, while using variable-size blocks (e.g., based on Rabin fingerprinting [3]) provides better deduplication efficiency.

1.1 Our Contributions

In this paper, we show how to design secure deduplication systems with higher reliability in cloud computing. We introduce the distributed cloud storage servers into deduplication systems to provide better fault tolerance. To further protect data confidentiality, the secret sharing technique is utilized, which is also compatible with the distributed storage systems. In more details, a file is first split and encoded into fragments by using the technique of secret sharing, instead of encryption mechanisms.

These shares will be distributed across multiple independent storage servers. Furthermore, to support deduplication, a short cryptographic hash value of the content will also be computed and sent to each storage server as the fingerprint of the fragment stored at each server. Only the data owner who first uploads the data is required to compute and distribute such secret shares, while all following users who own the same data copy do not need to compute and store these shares any more. To recover data copies, users must access a minimum number of storage servers through authentication and obtain the secret shares to reconstruct the data. In other words, the secret shares of data will only be accessible by the authorized users who own the corresponding data copy.

2 PROBLEM FORMULATION

2.1 System Model

This section is devoted to the definitions of the system model and security threats. Two kinds entities will be involved in this deduplication system, including the user and the storage cloud service provider (S-CSP). Both client-side deduplication and server-side deduplication are supported in our system to save the bandwidth for data uploading and storage space for data storing.

• *User*. The user is an entity that wants to outsource data storage to the S-CSP and access the data later. In a storage system supporting deduplication, the user only uploads unique data but does not upload any duplicate data to save the upload bandwidth. Furthermore, the fault tolerance is required by users in the system to provide higher reliability.

• *S-CSP*. The S-CSP is an entity that provides the outsourcing data storage service for the users. In the deduplication system, when users own and store the same content, the S-CSP will only store a single copy of these files and retain only unique data. A deduplication technique, on the other hand, can reduce the storage cost at the server side and save the upload bandwidth at the user side. For fault tolerance and confidentiality of data storage, we consider a quorum of S-CSPs, each being an independent entity. The user data is distributed across multiple S-CSPs. We deploy our deduplication mechanism in both file and block levels. Specifically, to upload a file, a user first performs the file-level duplicate check. If the file is a duplicate, then all its blocks must be duplicates as well, otherwise, the user further performs the blocklevel duplicate check and identifies the unique blocks to be uploaded. Each data copy (i.e., a file or a block) is associated with a *tag* for the duplicate check. All data copies and tags will be stored in the S-CSP.

2.2 Threat Model and Security Goals

Two types of attackers are considered in our threat model: (i) An outside attacker, who may obtain some knowledge of the data copy of interest via public channels.

An outside attacker plays the role of a user that interacts with the S-CSP; (ii) An inside attacker, who may have some knowledge of partial data information such as the ciphertext. An insider attacker is assumed to be honest-but-curious and will follow our protocol, which could refer to the S-CSPs in our system. Their goal is to extract useful information from user data. The following security requirements, including confidentiality, integrity, and reliability are considered in our security model.

Confidentiality. Here, we allow collusion among the SCSPs. However, we require that the number of colluded S-CSPs is not more than a predefined threshold. To this end, we aim to achieve data confidentiality against collusion attacks. We require that the data distributed and stored among the S-CSPs remains secure when they are unpredictable (i.e., have high min-entropy), even if the adversary controls a predefined number of S-CSPs. The goal of the adversary is to retrieve and recover the files that do not belong to them. This requirement has recently been formalized in [6] and called the privacy against chosen distribution attack. This also implies that the data is secure against the adversary who does not own the data. *Integrity*. Two kinds of integrity, including tag consistency and message authentication, are involved in the security model. Tag consistency check is run by the cloud storage server during the file uploading phase, which is used to prevent the duplicate/ciphertext replacement attack. If any adversary uploads a maliciously-generated ciphertext such that its tag is the same with another honestly-generated ciphertext, the cloud storage server can detect this dishonest behavior. Thus, the users do not need to worry about that their data are replaced and unable to be decrypted. Message authentication check is run by the users, which is used to detect if the downloaded and decrypted data are complete and uncorrupted or not. This security requirement is introduced to prevent the insider attack from the cloud storage service providers.

Reliability. The security requirement of reliability in deduplication means that the storage system can provide fault tolerance by using the means of redundancy. In more details, in our system, it can be tolerated even if a certain number of nodes fail. The system is required to detect and repair corrupted data and provide correct output for the users.

CONCLUSIONS

We proposed the distributed deduplication systems to improve the reliability of data while achieving the confidentiality of the users' outsourced data without an encryption mechanism. Four constructions were proposed to support file-level and fine-grained block-level data deduplication. The security of tag consistency and integrity were achieved.We implemented our deduplication systems using the Ramp secret sharing scheme and demonstrated that it incurs small encoding/decoding overhead compared to the network transmission overhead in regular upload/download operations.

REFERENCES

[1] Amazon, "Case Studies," https://aws.amazon.com/solutions/casestudies/#
backup.

[2] J. Gantz and D. Reinsel, "The digital universe in 2020: Big
data, bigger digi tal shadows, and biggest growth in the
far east," http://www.emc.com/collateral/analyst-reports/idcthe-
digital-universe-in-2020.pdf, Dec 2012.

[3] M. O. Rabin, "Fingerprinting by random polynomials," Center
for Research in Computing Technology, Harvard University, Tech.
Rep. Tech. Report TR-CSE-03-01, 1981.

[4] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer,
"Reclaiming space from duplicate files in a serverless distributed
file system." in *ICDCS*, 2002, pp. 617–624.

[5] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Dupless: Serveraided
encryption for deduplicated storage," in *USENIX Security*
Symposium, 2013.

[6] ——, "Message-locked encryption and secure deduplication," in
EUROCRYPT, 2013, pp. 296–312.

[7] G. R. Blakley and C. Meadows, "Security of ramp schemes," in
Advances in Cryptology: Proceedings of CRYPTO '84, ser. Lecture
Notes in Computer Science, G. R. Blakley and D. Chaum, Eds.
Springer-Verlag Berlin/Heidelberg, 1985, vol. 196, pp. 242–268.

[8] A. D. Santis and B. Masucci, "Multiple ramp schemes," *IEEE*
Transactions on Information Theory, vol. 45, no. 5, pp. 1720–1728,
Jul. 1999.

[9] M. O. Rabin, "Efficient dispersal of information for security, load
balancing, and fault tolerance," *Journal of the ACM*, vol. 36, no. 2,
pp. 335–348, Apr. 1989.

[10] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11,
pp. 612–613, 1979.

[11] J. Li, X. Chen, M. Li, J. Li, P. Lee, and W. Lou, "Secure deduplication
with efficient and reliable convergent key management," in
IEEE Transactions on Parallel and Distributed Systems, 2014, pp. vol.
25(6), pp. 1615–1625.

[12] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Proofs of ownership in remote storage systems." in *ACM Conference on Computer and Communications Security*, Y. Chen, G. Danezis, and V. Shmatikov, Eds. ACM, 2011, pp. 491–500.

[13] J. S. Plank, S. Simmerman, and C. D. Schuman, "Jerasure: A library in C/C++ facilitating erasure coding for storage applications - Version 1.2," University of Tennessee, Tech. Rep. CS-08-627, August 2008.

[14] J. S. Plank and L. Xu, "Optimizing Cauchy Reed-solomon Codes for fault-tolerant network storage applications," in *NCA-06: 5th IEEE International Symposium on Network Computing Applications*, Cambridge, MA, July 2006.

[15] C. Liu, Y. Gu, L. Sun, B. Yan, and D. Wang, "R-admad: High reliability provision for large-scale de-duplication archival storage systems," in *Proceedings of the 23rd international conference on Supercomputing*, pp. 370–379.

[16] M. Li, C. Qin, P. P. C. Lee, and J. Li, "Convergent dispersal: Toward storage-efficient security in a cloud-of-clouds," in *The 6th USENIX Workshop on Hot Topics in Storage and File Systems*, 2014.

[17] P. Anderson and L. Zhang, "Fast and secure laptop backups with encrypted de-duplication," in *Proc. of USENIX LISA*, 2010.

[18] Z. Wilcox-O'Hearn and B. Warner, "Tahoe: the least-authority filesystem," in *Proc. of ACM StorageSS*, 2008.

[19] A. Rahumed, H. C. H. Chen, Y. Tang, P. P. C. Lee, and J. C. S. Lui, "A secure cloud backup system with assured deletion and version control," in *3rd International Workshop on Security in Cloud Computing*, 2011.

[20] M. W. Storer, K. Greenan, D. D. E. Long, and E. L. Miller, "Secure data deduplication," in *Proc. of StorageSS*, 2008.

[21] J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl, "A secure data deduplication scheme for cloud storage," in *Technical Report*, 2013.

p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 13
September 2016

[22] D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Side channels in cloud services: Deduplication in cloud storage." *IEEE Security & Privacy*, vol. 8, no. 6, pp. 40–47, 2010.

[23] R. D. Pietro and A. Sorniotti, "Boosting efficiency and security in proof of ownership for deduplication." in *ACM Symposium on Information, Computer and Communications Security*, H. Y. Youm and Y. Won, Eds. ACM, 2012, pp. 81–82.

[24] J. Xu, E.-C. Chang, and J. Zhou, "Weak leakage-resilient client-side deduplication of encrypted data in cloud storage," in *ASIACCS*, 2013, pp. 195–206.

[25] W. K. Ng, Y. Wen, and H. Zhu, "Private data deduplication protocols in cloud storage." in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, S. Ossowski and P. Lecca, Eds. ACM, 2012, pp. 441–446.

Ms. R.Samatha rani received M.Tech degree from AuroraTechnological & Research Institute & Engineering , Warangal affiliated to JNTUH, Hydearabad. She is currently working as Assistant professor, Department of CSE, in Vinuthna Institute of Technology & Science ,Hasanparthy, Warangal, Telangana, India. Her interest includes Data Base Management Systems.

Ms. B.Vishwaja received B.Tech Degree from Vinuthna Institute of Technology & Science Hasanparthy, Warangal affiliated to KU, Warangal. She is currently pursuing M.Tech Degree in Software Engineering specialization in Vinuthna Institute of Technology & Science Hasanparthy, Warangal, Telangana, India